DEVELOPMENT AND APPLICATIONS OF ADAPTIVE IIR AND SUBBAND FILTERS

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF

Master of Technology
in
Telematics and Signal Processing

By

ARATI SAHOO

Department of Electronics and Communication Engineering
National Institute of Technology
Rourkela

2007

DEVELOPMENT AND APPLICATIONS OF ADAPTIVE IIR AND SUBBAND FILTERS

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF

Master of Technology
in
Telematics and Signal Processing

By

ARATI SAHOO

Under the Guidance of

Prof. G. Panda

Department of Electronics and Communication Engineering
National Institute of Technology
Rourkela

2007

National Institute Of Technology
Rourkela

CERTIFICATE

This is to certify that the thesis entitled, **“Development and Applications of Adaptive IIR and Subband Filters”** submitted by Ms **Arati Sahoo** in partial fulfillment of the requirements for the award of Master of Technology Degree in **Electronics & communication Engineering** with specialization in **“Telematics and Signal Processing”** at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by her under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Prof. G. Panda
Dept. of Electronics & Communication Engg.
National Institute of Technology
Rourkela-769008

Date:

ACKNOWLEDGEMENT

First of all, I would like to express my deep sense of respect and gratitude towards my advisor and guide **Prof (Dr.) Ganapati Panda**, who has been the guiding force behind this work. I want to thank him for introducing me to the field of Signal Processing and giving me the opportunity to work under him. I am greatly indebted to him for his constant encouragement and invaluable advice in every aspect of my academic life. His presence and optimism have provided an invaluable influence on my career and outlook for the future. I consider it my good fortune to have got an opportunity to work with such a wonderful person.

I express my respects to **Prof. (Dr.) K. K. Mahapatra, Prof. (Dr.) G. S. Rath, Prof. (Dr.) S.K. Patra, and Prof.(Dr.) S. Meher** for teaching me and also helping me how to learn. They have been great sources of inspiration to me and I thank them from the bottom of my heart.

I would like to thank all faculty members and staff of the Department of Electronics and Communication Engineering, N.I.T. Rourkela for their generous help in various ways for the completion of this thesis.

I would also like to mention the name of **Babita** for helping me a lot during the thesis period.

I would like to thank my classmates, especially **Ajit, Nihar** from whom I learned a lot and whose companionship I enjoyed so much during my stay at NIT, Rourkela.

I am especially indebted to my parents and brothers for their love, sacrifice, and support. They are my first teachers after I came to this world and have set great examples for me about how to live, study, and work.

Arati Sahoo
Roll No: 20507025
Dept of ECE
National Institute of Technology
Rourkela

CONTENTS

	Page No.
ABSTRACT	i
LIST OF FIGURES	ii
LIST OF TABLES	v
ABBREVIATIONS USED	vi
1. INTRODUCTION	1
1.1 Background	1
1.2 Motivation	2
1.3 Thesis Layout	4
2. ADAPTIVE IIR FILTER AND ITS APPLICAIONS	
2.1 Introduction	5
2.2 Adaptive filter problem	5
2.3 Adaptive IIR filter structure	6
2.4 Applications of adaptive filter	9
2.4.1 System identification	9
2.4.2 Channel equalization	13
2.5 Summary	14
3. ADAPTIVE IIR SYSTEM IDENTIFICATION	
3.1 Introduction	15
3.2 Adaptive model for IIR system identification	16
3.3 Different adaptive algorithms	22
3.3.1 Deterministic algorithms	22
3.3.1.1 General form of adaptive IIR algorithms	23
3.3.1.2 The Least-Mean-Square (LMS) algorithm	26

3.3.1.3 The Recursive-Least-Square (RLS) algorithm 32
3.3.2 Stochastic algorithm 37
3.3.2.1 Genetic algorithm 38
3.4 Results and discussion 46
3.4.1 LMS based system identification 46
3.4.2 RLS based system identification 49
3.4.3 GA based system identification 52
3.5 Summary 56

4. ADAPTIVE IIR CHANNEL EQUALIZATION

4.1 Introduction 57
4.2 Adaptive model for channel equalization 57
4.3 Nonminimum phase channel equalization 60
4.4 Results and discussion 64
4.5 Summary 65

5. SUBBAND ADAPTIVE FILTERING

5.1 Introduction 66
5.2 Multirate signal processing 68
5.2.1 Decimation and Interpolation 69
5.2.2 Polyphase decomposition 71
5.2.3 Two channel filter bank 72
5.3 Analysis of SAF for 2-band case 75
5.3.1 Structure of SAF 76
5.3.2 Adaptive algorithm 77
5.4 Result and discussion 78
5.5 Summary 81

6. CONCLUSION AND FUTURE SCOPE 82

REFERENCES 84

ABSTRACT

Adaptive infinite impulse response (IIR) filter is a challenging research area. Identifiers and Equalizers are among the most essential digital signal processing devices for digital communication systems. In this thesis, we consider adaptive IIR channel both for system identification and channel equalization purposes.

We focus on four different approaches: Least Mean Square (LMS), Recursive Least Square (RLS), Genetic Algorithm (GA) and Subband Adaptive Filter (SAF). The performance of conventional LMS and RLS based IIR system identification and channel equalization are compared. The RLS based IIR filter gives superior performance compared to its LMS counter part. It gives better matching between the desired output and estimated output as the order of the filter increases.

This thesis also examines enhanced structured stochastic global optimization algorithms for adaptive IIR filtering, with focus on an algorithm named Genetic Algorithm (GA). GA is based on the process of natural selection and does not require error gradient statistics. As a consequence, a GA is able to find a global error minimum. The GA is compared against the gradient descent training through extensive computations, where the GA demonstrates performance improvements. This training algorithm is given as an alternative training technique that overcomes the problems encountered by the gradient descent algorithm.

Subband adaptive filtering has attracted much attention in recent years. In this study, a subband structure is considered which is based upon the polyphase decomposition of filter to be adapted. This technique yield improved convergence rate when the number of bands in the filter is increased. The performance of subband adaptive filter is also compared with the conventional LMS.

Simulations results demonstrate that the adaptive IIR and subband filtering methods are directly applicable for large class of adaptive signal processing and communication systems.

Index Terms – Adaptive IIR filtering, system identification, channel equalization, subband filtering.

LIST OF FIGURES

1.1. Acoustic echo chancellor

2.1. The general adaptive filtering problem

2.2. Structure of an IIR filter

2.3. System identification

2.4. Inverse modeling

2.5. Linear prediction

3.1. Model for system identification

3.2. System identification configuration

3.3. Equation-error formulation

3.4. Output -error formulation

3.5. Single input digital IIR filter

3.6. Recursive adaptive filter

3.7. A GA iteration cycle

3.8. Biased roulette-wheel that is used in the selection of the mating pool

3.9 (a). Single point crossover

3.9 (b). Double point crossover

3.10. Mutation operation in GA

3.11. Performance study of 2nd order LMS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

3.12. Performance study of ^{3rd} order LMS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

3.13. Performance study of 4th order LMS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

3.14. Performance study of 5th order LMS based adaptive IIR filter

(a) Response match between desired and estimated
(b) NMSE plot in dB

3.15. Performance study of 2nd order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

3.16. Performance study of 3rd order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

3.17. Performance study of 4th order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

3.18. Performance study of 5th order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

3.19. Performance study of GA based 2nd order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

3.20. Performance study of GA based 3rd order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

3.21. Performance study of GA based 4th order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

3.22. Performance study of GA based 5th order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

4.1. A baseband communication System

4.2. A simple channel equalizer configuration

4.3. Channel equalization model

4.4. Proposed adaptive equalization System

4.5. Comparison of convergence performance between LMS based adaptive IIR and FIR equalizer

4.6. Comparison of convergence performance between LMS based and RLS based IIR Equalizer

5.1. Decomposition of analysis filter output

5.2. Decomposition of error signal

5.3. Decimation operation

5.4. Decimation for M = 2

5.5. Interpolation Operation

5.6. Interpolation for $M = 2$

5.7 (a). Filter followed by down sampler

5.7(b). Alternative Polyphase Network Presentations

5.8 (a). Upsampler followed by filter

5.8 (b). Polyphase decomposition

5.8 (c). Alternative polyphase network presentations

5.9. Two- Channel subband filter bank

5.10. SAF for 2-band case

5.11. Convergence performance of the SAF for M=2 with system noise absent (filter length L=80), No of training samples = 4,000

5.12. Convergence performance of the SAF for M=2 with system noise level of –20 dB (filter length L=80), No of training samples = 4,000

LIST OF TABLES

3.1: LMS algorithm for recursive filter

3.2: Update equation for RLS algorithm

3.3: Outline of the basic genetic algorithm

3.4: Simulation data for GA based 2nd order adaptive IIR filter

3.5: Simulation data for GA based 3rd order adaptive IIR filter

3.6: Simulation data for GA based 4th order adaptive IIR filter

3.7: Simulation data for GA based 5th order adaptive IIR filter

3.8: Comparison of computational complexity for different algorithms

5.3: Decimation operation

ABBREVIATIONS USED

IIR	Infinite Impulse Response
FIR	Finite Impulse Response
GA	Genetic Algorithm
LMS	Least Mean Square
RLS	Recursive Least Square
SAF	Subband Adaptive Filter
EEG	Electro Encephalo Gram
ISI	Inter Symbol Interference
SHARF	Simply Hyperstable Recursive Filter
MSE	Mean Square Error
NMSE	Normalized Mean Square Error

Chapter 1

INTRODUCTION

Background

Motivation

Thesis Layout

CHAPTER 1

1. INTRODUCTION

1.1 Background

Over the last several years, adaptive infinite- impulse-response (IIR) filtering has been an active area of research, and it has been considered for a variety of applications in signal processing and communications. Examples of some important applications include system identification, channel equalization, linear prediction, echo cancellation, and adaptive array processing. The primary advantage of an adaptive IIR filter is that it can provide significantly better performance than an adaptive FIR filters having the same number of coefficients. This is because the desired response can be approximated more efficiently by the output of a filter that has both poles and zeros compared to one that has only zeros.

In telephone communication a common problem encountered is the presence of echo, which is produced when the signal passes through telephone channels. Removal of this echo requires precise knowledge of the channel, which may be time varying. This calls for adaptive estimation of the channel, which is characterized by time varying impulse response. In recent years, there has been a marked interest in the application of adaptive filtering to system identification and channel equalization where the impulse response involved is long. We briefly describe this application to motivate our study.

A Genetic Algorithm (GA) is a stochastic training scheme that need not have a derivation that requires knowledge of the local error gradient [1.1], which gradient-descent training relies on. A GA consists of an evolutional process that raises the fitness of a population using the Darwinian survival of the fittest criterion. A GA relies upon the use of a solution population. Each solution within the population has to generate a cost value in each training iteration. GAs has proven to be useful in training and search applications that suffer from stability problems, locating solutions that have previously been unobtainable for IIR filters [1.2].

In hands-free telephone sets and teleconferencing systems, both ends of the telephone line consist of audio terminals in Fig. 1.1. The received speech signal $y(n)$ is fed to a loudspeaker (LS), which radiates acoustic waves. These waves are fed back to the remote user through the microphone (MP) and constitute the so-called echo. To cancel this echo, we take a sample of $y(n)$, modify it by passing it through an adaptive filter $S(z)$, and subtract the resulting signal from $z(n)$. $S(z)$ is the estimate of the impulse response of the path that the signal *y(n)* takes to form the echo *z(n)*. For complete cancellation of this echo, the impulse response of the adaptive filter *S(z)* may have to be very long [1.3].

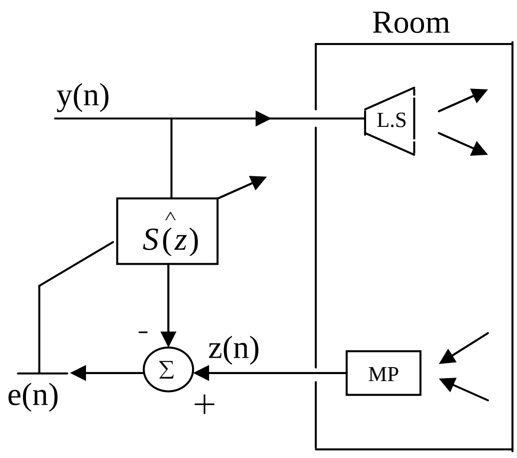


Fig. 1.1 Acoustic echo chancellor

In the above example, the adaptation of the filter is based on the error signal *e(n)*. The algorithm used for adaptation is generally a gradient type [1.4]. The least mean square (LMS) algorithm of Widrow *et al* [1.3] has been used widely in such applications. However, it suffers from slow convergence when the input signal to the adaptive filter is correlated, which is generally the case in the above problem. Generally communication channels are not linear, most of the times these are corrupted with noise. The performance of LMS algorithm is very poor if any Gaussian noise is added.

In this thesis we present adaptive IIR filtering using some adaptive algorithm e.g. Recursive Least square Algorithm (RLS) and Genetic Algorithm (GA) which have faster convergence rate and subband adaptive filtering (SAF) technique for the estimation of such channels. The SAF structure uses polyphase decomposition. To prevent distortion due to aliasing and imaging perfect reconstruction filters are used which are real.

1.2 Motivation

Adaptive filtering has a tremendous application in the field of signal processing and communications such as system identification, channel equalization, linear prediction, and noise cancellation etc. The most general adaptive IIR algorithms require that the filter is operating in system identification configuration such that the unknown system can be represented by a stable rational transfer function.

In system identification, the adaptive filter attempts to iteratively determine an optimal model for the unknown system, or "plant", based on some function of the error between the output of the adaptive filter and the output of the plant. The optimal model or solution is attained when this function of the error is minimized. The adequacy of the resulting model depends on the structure of the adaptive filter, the algorithm used to update the adaptive filter parameters, and the characteristics of the input signal.

When the parameters of a physical system are not available or time dependent it is difficult to obtain the mathematical model of the system. In such situations, the system parameters should be obtained using a system identification procedure. The purpose of system identification is to construct a mathematical model of a physical system from input/output. Studies on linear system identification have been carried out for more than three decades [1.6, 1.7]. However, identification of IIR systems is a promising research area

Adaptive equalization is used to undo the effects of a communication channel or produce an inverse model of an unknown plant. For adaptive IIR equalization, instability occurs as poles move outside of the unit circle. Such problems are overcome by using different equalization techniques. Hence IIR channel estimation is a key problem in communication system. Several approaches and SAF have been discussed recently for estimation of IIR systems.

1.3 Thesis Layout

In chapter2, the adaptive filter problem and the adaptive IIR filter structure are introduced. In this various applications of adaptive filters and their structures are discussed briefly.

In chapter 3, the theory and model of adaptive IIR system identification are discussed. We focus on different adaptive algorithms such as deterministic and stochastic algorithms. In deterministic approach, we discussed the Least Mean Square (LMS) and Recursive Least Square (RLS) algorithms. And in stochastic approach, the evolutionary computing technique Genetic Algorithm (GA) is derived. Simulation results are carried out for comparison of conventional LMS based IIR system with RLS based system identification under different order of the filter and noise conditions. A brief introduction to evolutionary computing technique and its simulation study is given.

In chapter 4, the model of adaptive IIR channel equalization and its instability problem occurs are discussed. In order to overcome the instability problem occur in this, an adaptive IIR equalizer for nonminimum-phase channel is introduced. The performance comparison study of LMS based IIR equalizer and RLS based IIR equalizer are given and also the comparison result of IIR equalizer and FIR equalizer is discussed.

In Chapter 5, a new structure is presented for the subband adaptive filter (SAF) and a new criterion for the adaptation algorithm that results in significant improvement in the convergence rate when LMS algorithm used for adaptation is outlined. The structure exploits the polyphase decomposition of the adaptive filter. To prevent any distortion such as aliasing and imaging we use perfect reconstruction filter banks. Simulations are carried for 2-band and full band case.

In Chapter 6, we discuss the conclusion and the future works that can be done.

*****

Chapter 2

ADAPTIVE IIR FILTER AND ITS APPLICATIONS

Adaptive filter problem

Adaptive IIR filter structure

Applications of adaptive filter

CHAPTER 2

2. ADAPTIVE IIR FILTER AND ITS APPLICATIONS

2.1 Introduction

Infinite Impulse Response (IIR) is an important property of digital signal processing systems. They have an impulse response function which is non-zero over an infinite length of time. This is in contrast to finite impulse response filters (FIR) which have fixed-duration impulse responses [2.1]. The essential and principal property of an adaptive system is its time-varying, self-adjusting performance. An adaptive filter is defined by four aspects [2.2]:

1. the *signals* being processed by the filter
2. the *structure* that defines how the output signal of the filter is computed from its input signal
3. the *parameters* within this structure that can be iteratively changed to alter the filter's input-output relationship
4. the *adaptive algorithm* that describes how the parameters are adjusted from one time instant to the next.

Adaptive infinite-impulse-response (IIR) filters are contemplated as replacements for adaptive finite impulse-response (FIR) filters when the desired filter can be more economically modeled with poles and zeros than with the all-zero form of an FIR tapped-delay line. It has considered for variety applications in adaptive signal processing and communications e.g. system identification, channel equalization, adaptive array processing, linear prediction, line enhancer etc.

2.2 Adaptive filter problem

Figure 2.1 shows a block diagram in which a sample from a digital input signal $x(n)$ is fed into a device, called an adaptive filter [2.4], that computes a corresponding output signal sample $y(n)$ at time n. For the moment, the structure of the adaptive filter is not important, except for the fact that it contains adjustable parameters whose values affect how $y(n)$ is computed. The output signal is compared to a second signal $d(n)$, called the desired response signal, by subtracting the two samples at time n. This difference signal, given by

$$e(n) = d(n) - y(n) \tag{2.1}$$

is known as the error signal. The error signal is fed into a procedure that alters or adapts the parameters of the filter from time n to time $(n + 1)$ in a well-defined manner. This process of adaptation is represented by the oblique arrow that pierces the adaptive filter block in the figure.

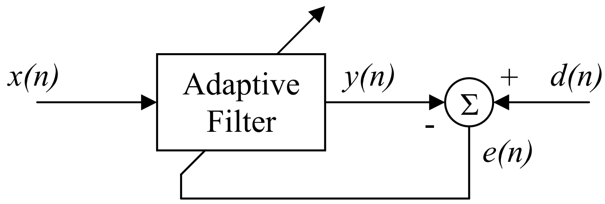


Fig 2.1: The general adaptive filtering problem.

As the time index n is incremented, it is hoped that the output of the adaptive filter becomes a better and better match to the desired response signal through this adaptation process, such that the magnitude of $e(n)$ decreases over time. This is specified by the form of the adaptive algorithm used to adjust the parameters of the adaptive filter. In the adaptive filtering task, adaptation refers to the method by which the parameters of the system are changed from time index n to time index $(n + 1)$. The number and types of parameters within this system depend on the computational structure chosen for the system. We now discuss different filter structures that have been proven useful for adaptive filtering task

2.3 Adaptive IIR filter structure

Traditionally, adaptive signal processing has, for the most part, been carried out using Finite Impulse Response (FIR) filters. The unimodal property of their mean square error surfaces allows adaptive algorithms based on gradient search techniques to be applied. Additionally, stability of the FIR filter can be guaranteed during adaptation. Infinite Impulse Response (IIR) filters are generally less computationally expensive due to their recursive nature and thus give better performance for a given order of filter [2.5]. In addition stability is also a concern with adaptive IIR filters, which is a challenging research area.

Two approaches have been taken in IIR filter adaptation, the equation-error and output-error formulations [2.6]. The equation-error formulation essentially renders the problem of adaptation one of updating two FIR filters. Consequently, well known adaptive FIR filter algorithms can be used in the search for the unique stationary point. This formulation, however, has the disadvantage that in the presence of noise or incomplete modeling, biased estimates of the filter coefficients may be produced. The output-error formulation updates the feedback coefficients of the IIR filter directly in the pole-zero form. The multi-modal nature of the mean square error surface may result in adaptive algorithms utilizing gradient search becoming stuck in a local minimum. Stability is also a concern with adaptive IIR filters. Should one of the poles be updated outside the unit circle and remain there for a significant period of time, the filter may become unstable, causing the output to grow without bound. Unfortunately, stability monitoring is computationally expensive and generally not robust for the conventional direct form implementation [2.7].

In general, any system with a finite number of parameters that affect how *y(n)* is computed from *x(n)* could be used for the adaptive filter in Fig. 2.1. Define the parameter or coefficient vector *W(n)* as

$$W(n) = [w_0(n) w_1(n) \ldots w_{L-1}(n)]^T \qquad (2.2)$$

where $\{w_i(n)\}$, $0 < i < L - 1$ are the L parameters of the system at time n. With this definition, we could define a general input-output relationship for the adaptive filter as

$$y(n)=f(W(n),y(n-1),y(n-2),\ldots,y(n-N),x(n),x(n-1),\ldots,x(n-M+1)) \qquad (2.3)$$

where $f(\,)$ represents any well-defined linear or nonlinear function and M and N are positive integers.

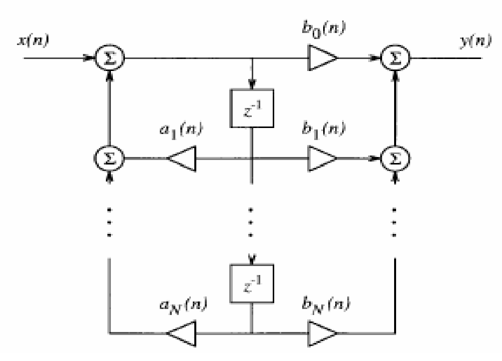


Fig 2.2: Structure of an IIR filter

.

The structure of a direct-form IIR filter is shown in Fig. 2.2. In this case, the output of the system can be mathematically represented as

$$y(n) = \sum_{i=1}^{N} a_i(n) y(n-i) + \sum_{j=0}^{N} b_j(n) x(n-j) \qquad (2.4)$$

Thus, for purposes of computing the output signal *y{n),* the IIR structure involves a fixed number of multiplies, adds, and memory locations not unlike the direct-form FIR structure. A critical issue in the choice of an adaptive filter's structure is its computational complexity. Since the operation of the adaptive filter typically occurs in real time, all of the calculations for the system must occur during one sample time. The structures described above are all useful because *y(n)* can be computed in a finite amount of time using simple arithmetical operations and finite amounts of memory.

When considering the adaptive filter problem as illustrated in Fig. 2.1 for the first time, a reader is likely to ask, "If we already have the desired response signal, what is the point of trying to match it using an adaptive filter?" In fact, the concept of "matching" *y(n)* to *d(n)* with some system obscures the subtlety of the adaptive filtering task. Consider the following issues that pertain to many adaptive filtering problems:

- *In practice, the quantity of interest is not always d(n).* Our desire may be to represent in *y(n)* a certain component of *d(n)* that is contained in *x{n),* or it may be to isolate a component of *d(n)* within the error *e(n)* that is *not* contained in x(n). Alternatively, we may be solely interested in the values of the parameters in *W(n)* and have no concern about *x{n), y{n),* or *d(n)* themselves. Practical examples of each of these scenarios are provided later in this chapter.
- There are situations in which d(n) is not available at all times. In such situations, adaptation typically occurs only when *d(n)* is available. When *d(n)* is unavailable, we typically use our most-recent parameter estimates to compute *y (n)* in an attempt to estimate the desired response signal *d(n).*
- There are real-world situations in which d(n) is never available. In such cases, one can use additional information about the characteristics of a "hypothetical" *d(n),* such as its predicted statistical behavior or amplitude characteristics, to form suitable estimates of *d(n)* from the signals available to the adaptive filter. Such methods are collectively called blind adaptation algorithms. The fact that such schemes even work is a tribute both to the ingenuity of the developers of the algorithms and to the technological maturity of the adaptive filtering field.

It should also be recognized that the relationship between *x(n)* and *d(n)* can vary with time. In such situations, the adaptive filter attempts to alter its parameter values to follow the changes in this relationship as "encoded" by the two sequences *x(n)* and *d(n).* This behavior is commonly referred to as tracking.

2.4 Applications of adaptive IIR filters

Perhaps the most important driving forces behind the developments in adaptive filters throughout their history have been the wide range of applications in which such systems can be used. We now discuss the forms of these applications in terms of more-general problem classes that describe the assumed relationship between *d(n)* and *x(n).* Our discussion illustrates the key issues in selecting an adaptive filter for a particular task [2.8].

2.4.1 System identification

Consider Fig. 2.3, which shows the general problem of system identification. In this diagram, the system enclosed by dashed lines is a "black box," meaning that the quantities inside are not observable from the outside. Inside this box is (1) an unknown system which represents a general input-output relationship and (2) the signal $\eta_i(n)$, called the observation noise signal because it corrupts the observations of the signal at the output of the unknown system.

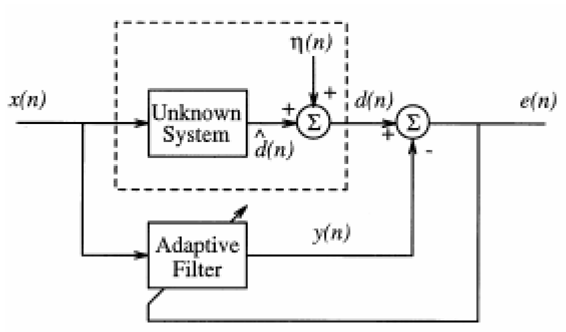


Fig 2.3: System identification.

Let *d(n)* represent the output of the unknown system with *x(n)* as its input. Then, the desired response signal in this model is

$$d(n) = \hat{d}(n) + \eta(n) \tag{2.5}$$

Here, the task of the adaptive filter is to accurately represent the signal *d(n)* at its output. If *y(n) = d (n)*, then the adaptive filter has accurately modeled or identified the portion of the unknown system that is driven by *x{n)*.

Since the model typically chosen for the adaptive filter is a linear filter, the practical goal of the adaptive filter is to determine the best linear model that describes the input-output relationship of the unknown system. Such a procedure makes the most sense when the unknown system is also a linear model of the same structure as the adaptive filter, as it is possible that $y(n) = d\{n)$ for some set of adaptive filter parameters. For ease of discussion, let the unknown system and the adaptive filter both be IIR filters, such that

$$d(n) = W_{OPT}^{T}(n)X(n) + \eta(n), \tag{2.6}$$

where $W_{opt}(n)$ is an optimum set of filter coefficients for the unknown system at time n. In this problem formulation, the ideal adaptation procedure would adjust $W(n)$ such that $W(n) = W_{opt}(n)$ as $n \to \infty$. In practice, the adaptive filter can only adjust $W(n)$ such that $y(n)$ closely approximates $d(n)$ over time.

The system identification task is at the heart of numerous adaptive filtering applications. We list several of these applications here.

Channel identification

In communication systems, useful information is transmitted from one point to another across a medium such as an electrical wire, an optical fiber, or a wireless radio link. Nonidealities of the transmission medium or channel distort the fidelity of the transmitted signals, making the deciphering of the received information difficult. In cases where the effects of the distortion can be modeled as a linear filter, the resulting "smearing" of the transmitted symbols is known as inter-symbol interference (ISI). In such cases, an adaptive filter can be used to model the effects of the channel ISI for purposes of deciphering the received information in an optimal manner. In this problem scenario, the transmitter sends to the receiver a sample sequence $x(n)$ that is known to both the transmitter and receiver. The receiver then attempts to model the received signal $d(n)$ using an adaptive filter whose input is the known transmitted sequence $x(n)$. After a suitable period of adaptation, the parameters of the adaptive filter in $W(n)$ are fixed and then used in a procedure to decode future signals transmitted across the channel. Channel identification is typically employed when the fidelity of the transmitted channel is severely compromised or when simpler techniques for sequence detection cannot be used.

Plant identification

In many control tasks, knowledge of the transfer function of a linear plant is required by the physical controller so that a suitable control signal can be calculated and applied. In such cases, we can characterize the transfer function of the plant by exciting it with a known signal $x(n)$ and then attempting to match the output of the plant $d(n)$ with a linear adaptive filter. After a suitable period of adaptation, the system has been adequately modeled, and the resulting adaptive filter coefficients in $W(n)$ can be used in a control scheme to enable the overall closed-loop system to behave in the desired manner.

In certain scenarios, continuous updates of the plant transfer function estimate provided by W(n) are needed to allow the controller to function properly. A discussion of these adaptive control schemes and the subtle issues in their use is given in [2.9].

Echo cancellation for long-distance transmission

In voice communication across telephone networks, the existence of junction boxes called hybrids near either end of the network link hampers the ability of the system to cleanly transmit voice signals. Each hybrid allows voices that are transmitted via separate lines or channels across a long-distance network to be carried locally on a single telephone line, thus lowering the wiring costs of the local network. However, when small impedance mismatches between the long distance lines and the hybrid junctions occur, these hybrids can reflect the transmitted signals back to their sources, and the long transmission times of the long-distance network—about 0.3 s for a trans-oceanic call via a satellite link—turn these reflections into a noticeable echo that makes the understanding of conversation difficult for both callers. The traditional solution to this problem prior to the advent of the adaptive filtering solution was to introduce significant loss into the long-distance network so that echoes would decay to an acceptable level before they became perceptible to the callers. Unfortunately, this solution also reduces the transmission quality of the telephone link and makes the task of connecting long distance calls more difficult.

An adaptive filter can be used to cancel the echoes caused by the hybrids in this situation. Adaptive niters are employed at each of the two hybrids within the network. The input x(n) to each adaptive filter is the speech signal being received prior to the hybrid junction, and the desired response signal d(n) is the signal being sent out from the hybrid across the long-distance connection. The adaptive filter attempts to model the transmission characteristics of the hybrid junction as well as any echoes that appear across the long-distance portion of the network. When the system is properly designed, the error signal e(n) consists almost totally of the local talker's speech signal, which is then transmitted over the network. Such systems were first proposed in the mid-1960s and are commonly used today.

Acoustic echo cancellation

A related problem to echo cancellation for telephone transmission systems is that of acoustic echo cancellation for conference-style speakerphones. When using a speakerphone, a caller would like to turn up the amplifier gains of both the microphone and the audio loudspeaker in order to transmit and hear the voice signals more clearly. However, the feedback path from the device's loudspeaker to its input microphone causes a distinctive howling sound if these gains are too high. In this case, the culprit is the room's response to the voice signal being broadcast by the speaker; in effect, the room acts as an extremely poor hybrid junction, in analogy with the echo cancellation task discussed previously. A simple solution to this problem is to only allow one person to speak at a time, a form of operation called half-duplex transmission. However, studies have indicated that half-duplex transmission causes problems with normal conversations, as people typically overlap their phrases with others when conversing.

To maintain full-duplex transmission, an acoustic echo canceller is employed in the speakerphone to model the acoustic transmission path from the speaker to the microphone. The input signal *x(n)* to the acoustic echo canceller is the signal being sent to the speaker, and the desired response signal *d(n)* is measured at the microphone on the device. Adaptation of the system occurs continually throughout a telephone call to model any physical changes in the room acoustics. Such devices are readily available in the marketplace today. In addition, similar technology can and is used to remove the echo that occurs through the combined radio/room/telephone transmission path when one places a call to a radio or television talk show.

Adaptive noise canceling

When collecting measurements of certain signals or processes, physical constraints often limit our ability to cleanly measure the quantities of interest. Typically, a signal of interest is linearly mixed with other extraneous noises in the measurement process, and these extraneous noises introduce unacceptable errors in the measurements. However, if a linearly related reference version of any one of the extraneous noises can be cleanly sensed at some other physical location in the system, an adaptive filter can be used to determine the relationship between the noise reference *x(n)* and the component of this noise that is contained in the measured signal *d{n)*. After adaptively subtracting out this component, what remains in *e(n)* is the signal of interest. If several extraneous noises corrupt the measurement of interest, several adaptive filters can be used in parallel as long as suitable noise reference signals are available within the system [2.3].

Adaptive noise canceling has been used for several applications. One of the first was a medical application that enabled the electroencephalogram (EEG) of the fetal heartbeat of an unborn child to be cleanly extracted from the much-stronger interfering EEG of the maternal heartbeat signal.

2.4.2 Channel equalization

Inverse modeling

We now consider the general problem of inverse modeling, as shown in Fig. 2.4. In this diagram, a source signals *s(n)* is fed into an unknown system that produces the input signal *x(n)* for the adaptive filter. The output of the adaptive filter is subtracted from a desired response signal that is a delayed version of the source signal, such that

$$d(n) = s(n - \Delta) \tag{2.7}$$

where Δ is a positive integer value. The goal of the adaptive filter is to adjust its characteristics such that the output signal is an accurate representation of the delayed source signal.

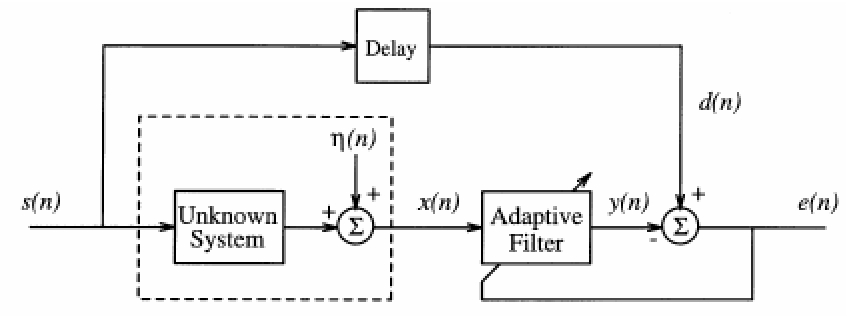


Fig 2.4 Inverse Modeling

Channel equalization is an alternative to the technique of channel identification described previously for the decoding of transmitted signals across nonideal communication channels. In both cases, the transmitter sends a sequence *s(n)* that is known to both the transmitter and receiver. However, in equalization, the received signal is used as the input signal *x(n)* to an adaptive filter, which adjusts its characteristics so that its output closely matches a delayed version *s(n*-Δ) of the known transmitted signal. After a suitable adaptation period, the coefficients of the system either are fixed and used to decode future transmitted messages or are adapted using a crude estimate of the desired response signal that is computed from *y{n)*. This latter mode of operation is known as decision-directed adaptation.

Channel equalization was one of the first applications of adaptive filters and is described in the pioneering work of Lucky [2.10]. Adaptive equalization is also useful for wireless communication systems. Qureshi [2.11] provides a tutorial on adaptive equalization.

Inverse plant modeling

In many control tasks, the frequency and phase characteristics of the plant hamper the convergence behavior and stability of the control system. We can use a system of the form in Fig. 2.4 to compensate for the nonideal characteristics of the plant and as a method for adaptive control. In this case, the signal *s(n)* is sent at the output of the controller, and the signal *x(n)* is the signal measured at the output of the plant. The coefficients of the adaptive filter are then adjusted so that the cascade of the plant and adaptive filter can be nearly represented by the pure delay $z^{-\Delta}$ [2.12].

Adaptive line enhancement

In some situations, the desired response signal *d(n)* consists of a sum of a broadband signal and a nearly periodic signal, and it is desired to separate these two signals without specific knowledge about the signals (such as the fundamental frequency of the periodic component).In these situations, an adaptive filter configured as in Fig. 2.5 can be used. For this application, the delay Δ is chosen to be large enough such that the broadband component in *x(n)* is uncorrelated with the broadband component in *x(n*-Δ).

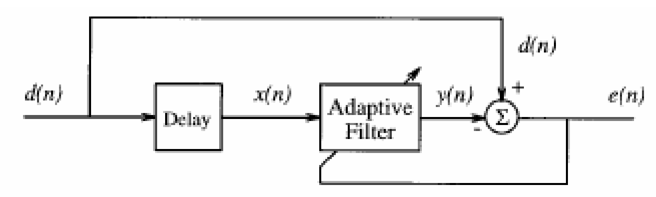


Fig 2.5 Linear Prediction

In this case, the broadband signal cannot be removed by the adaptive filter through its operation, and it remains in the error signal *e(n)* after a suitable period of adaptation. The adaptive filter's output *y(n)* converges to the narrowband component, which is easily predicted given past samples. The name line enhancement arises because periodic signals are characterized by lines in their frequency spectra, and these spectral lines are enhanced at the output of the adaptive filter.

2.5 Summary

This chapter has given an overview of the important structures and applications used in adaptive IIR filtering. Adaptive IIR filter has many applications such as system identification; channel equalization, linear prediction, adaptive line enhancement etc. are discussed briefly.

*****

Chapter 3

ADAPTIVE IIR SYSTEM IDENTIFICATION

Adaptive model for IIR system identification

Different adaptive algorithms

Results and discussion

CHAPTER 3

3. ADAPTIVE IIR SYSTEM IDENTIFICATION

3.1 Introduction

System identification is the experimental approach to process modeling. System identification includes the following steps

Experiment design: Its purpose is to obtain good experimental data and it includes the choice of the measured variables and of the character of the input signals.

Selection of model structure: A suitable model structure is chosen using prior knowledge and trial and error.

Choice of the criterion to fit: A suitable cost function is chosen, which reflects how well the model fits the experimental data.

Parameter estimation: An optimization problem is solved to obtain the numerical values of the model parameters.

Model validation: The model is tested in order to reveal any inadequacies.

The key problem in system identification is to find a suitable model structure, within which a good model is to be found. Fitting a model within a given structure (parameter estimation) is in most cases a lesser problem. A basic rule in estimation is not to estimate what you already know. In other words, one should utilize prior knowledge and physical insight about the system when selecting the model structure. It is customary to distinguish between three levels of prior knowledge, which have been color-coded as follows.

1. White Box models: This is the case when a model is perfectly known; it has been possible to construct it entirely from prior knowledge and physical insight.

2. Grey Box models: This is the case when some physical insight is available, but several parameters remain to be determined from observed data. It is useful to consider two sub cases- Physical Modeling: A model structure can be built on physical grounds, which has a certain number of parameters to be estimated from data.

- Semi-physical modeling: Physical insight is used to suggest certain nonlinear combinations of measured data signal. These new signals are then subjected to model structures of black box character.

3. Black Box models**:** No physical insight is available or used, but the chosen model structure belongs to families that are known to have good flexibility and have been "successful in the past".

Basic techniques for estimating the parameters in the structures are criterion minimization, as well as two step procedures, where first the relevant basis functions are determined, using data, and then a linear least squares step to determine the coordinates of the function approximation. A particular problem is to deal with the large number of potentially necessary parameters. This is handled by making the number of "used" parameters considerably less than the number of "offered" parameters, by regularization, shrinking, pruning or regressor selection.

In this chapter, we present the general basic system identification problem, solution via adaptive approach, stochastic approach and introduce the mathematical notation for representing the form and operation of the adaptive filter. We then discuss several different linear models that have been proven to be useful in practical applications for IIR channels. We provide an overview of the many and varied applications in which adaptive filters have been successfully used. We give a simple derivation of the least-mean-square (LMS) algorithm*,* which is perhaps the most popular method for adjusting the coefficients of an adaptive filter, and we discuss some of this algorithm's properties and shortcomings in Section 3.3. We discuss recursive LMS algorithm & its limitation, then the simply hyperstable recursive filter (SHARF). Finally, we discuss new algorithms and techniques, which can be applied in place of conventional methods such as Genetic Algorithm. Simulation results are given for the described algorithm to show its performance.

3.2 Adaptive model for IIR system identification

Consider Fig. 3.1, which shows the general problem of system identification. In this diagram, the system enclosed by dashed lines is a "black box" , [2.1] meaning that the quantities inside are not observable from the outside. Inside this box, there is an unknown system, which represents a general input-output relationship. In many practical cases, the plant to be modeled is noisy, that is, has internal random disturbing forces. In our problem it is represented by the signal η *(n),* called the observation noise. Internal plant noise appears at the plant output and is commonly represented there as an adaptive noise. This noise is generally uncorrelated with the plant input. If this is the case and if the adaptive model weights are adjusted to minimize mean-square error, it can be shown that the least square solutions will be unaffected by the presence of plant noise. This is difficult to say that the convergence of the adaptive process will be unaffected by plant noise, only that signal because it corrupts the observations of the signal at the output of the unknown system. The expected weight vector of the adaptive model after convergence will be unaffected.

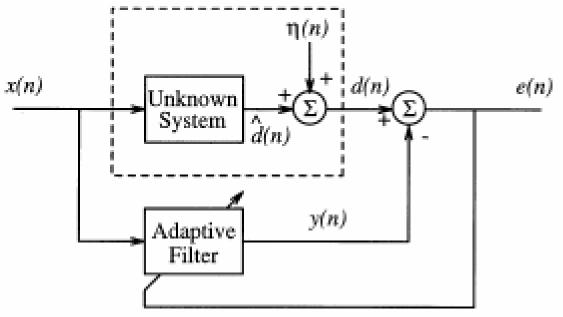


Fig. 3.1 Model for system identification

Let *d (n)* represent the output of the unknown system with *x (n)* as its input. Then, the desired response signal in this model is $d(n) = \hat{d}(n) + \eta(n)$.

Here, the task of the adaptive filter is to accurately represent the signal *d (n)* at its output. If *y (n) = d (n)*, then the adaptive filter has accurately modeled or identified the portion of the unknown system that is driven by *x (n)*.

For our problem we have assumed that the input signal *x (n)* and noise signal $\eta(n)$, are mutually uncorrelated white random sequences with zero mean. And hence

$$E\left[x_n^2\right]=\tfrac{1}{12}, \qquad E\left[\eta_n^2\right]=\tfrac{1}{12} \tag{3.1}$$

The white noise is Gaussian in nature having probability density function as follows:

$$p(\delta) = \frac{1}{\sqrt{2\pi\sigma_\delta^{\ 2}}}\exp\left(\frac{-(\delta-m)^2}{2\sigma_\delta^{\ 2}}\right) \tag{3.2}$$

As the adaptation process reaches Wiener solution, the power of error signal will be exactly equal to the extra noise added. It is not exactly required that the impulse response of both unknown plant and adaptive model shown in Fig. 3.1 should match, but the desired output and estimated output should match.

Fig. 3.2 shows the adaptive filter in a system identification configuration where θ_* are the unknown system parameters, and *d(n)* is simply the measured output of the system, which usually includes an additive noise process *v(n)*. The objective of the algorithm is to minimize a performance criterion that is based on the prediction error *e(n)* (sometimes called the estimation error), defined by *e(n) = d(n) -y(n).*

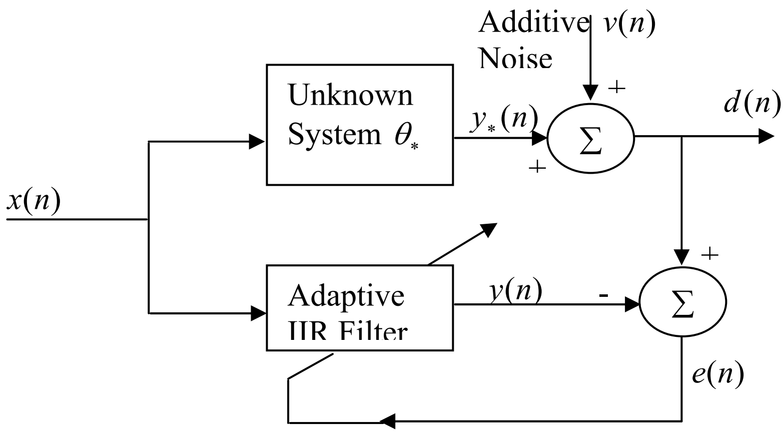


Fig.3.2 System identification configuration

Fundamentally, there have been two approaches to adaptive IIR filtering that correspond to different formulations of the prediction error; these are known as equation error and output error methods. In the equation-error formulation [3.1, 3.2] the feedback coefficients of an IIR filter are updated in an all-zero, nonrecursive form, which are then copied to a second filter implemented in an all-pole form. This formulation is essentially a type of adaptive FIR (finite-impulse-response) filtering, and the corresponding algorithms have properties that are well understood and predictable. Unfortunately, the equation-error approach can lead to biased estimates of the coefficients θ_*. The output-error formulation [3.3] updates the feedback coefficients directly in a pole-zero, recursive form and it does not generate biased estimates. However, the adaptive algorithms can converge to a local minimum of ξ leading to an incorrect estimate of θ_* and their convergence properties are not easily predicted. As a result, there is a trade-off between converging to a biased estimate of the coefficients and converging to a local minimum of ξ.

Equation-error and output-error formulation

Equation-error formulation

Consider the equation-error adaptive IIR filter shown in Fig. 3.3, which is characterized by the nonrecursive difference equation:

$$y_e(n) = \sum_{m=1}^{N-1} a_m(n)d(n-m) + \sum_{m=0}^{M-1} b_m(n)x(n-m) \tag{3.3}$$

where $\{a_m(n), b_m(n)\}$ are the adjustable coefficients and the subscript '*e*' is used to distinguish this output from that of the output-error formulation. Observe that (3.3) is a two-input, single-output filter that depends on delayed samples of the input $x(n - m)$, $m = 0, \ldots, M - 1$, and of the desired response $d(n - m)$, $m = 1, \ldots, N - 1$. It does not depend on delayed samples of the output and, therefore, the filter does not have feedback; the output is clearly a linear function of the coefficients. This property greatly simplifies the derivation of gradient-based algorithms. Since $d(n)$ and $x(n)$ are not functions of the coefficients, the derivative of $y_e(n)$ with respect to the coefficients is nonrecursive and is easy to compute.

This expression can be rewritten in a more convenient form using delay-operator notation as follows:

$$y_e(n) = A(n,q)d(n) + B(n,q)x(n) \tag{3.4}$$

where the polynomials in q represent time-varying filters and are defined by

$$A(n,q) = \sum_{m=1}^{N-1} a_m(n)q^{-m} \quad \text{and} \quad B(n,q) = \sum_{m=0}^{M-1} b_m(n)q^{-m} \tag{3.5}$$

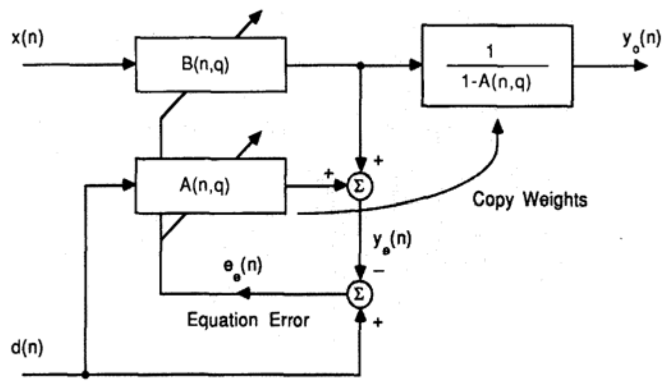


Fig 3.3 Equation-error formulation.

Note that the lower limit of the sum for A(n,q) begins with $m = 1$; consequently, A(n,q)d(n) depends only on delayed samples of *d.* The argument *n* emphasizes the time dependence of the coefficients and *q-1* is the delay operator such that $q^{-m}x(n) = x(n-m)$. These functions of *q* operate on time signals only from the left as in (3.4). By replacing *q* with the complex variable z, the expressions in (3.5) become z-transforms (transfer functions), assuming that the coefficients are fixed (independent of time), i.e., $a_m(n) \to a_m$, and $b_m(n) \to b_m$, so that

$A(n,q) \to A(z)$ and $B(n,q) \to B(z)$. This form can be used to find the zeros of the adaptive filter at any instant of time. For example, after each update of the coefficients and before the coefficients *{$a_m(n)$}* are copied to the inverse filter (Fig. 3.3), it will be necessary to monitor the zeros of *1 - A(z)* to determine if its inverse is a stable system. If it is not stable, then some method of projecting the roots inside the unit circle will be necessary before the inverse filter is formed.

The equation error is given by *e(n) = d(n) - $y_e(n)$*. It is called this because it is generated by subtracting two difference equations: *[1 – A (n, q)] d (n)* and *B (n, q) x (n)*. Note that *$e_e(n)$* is also a linear function of the coefficients; as a result, the mean-square-equation error (MSEE) is a quadratic function with a single global minimum (provided the data correlation matrix is nonsingular) and no local minima [3.4]. In many ways, the performance of an equation-error adaptive IIR filter is like that of an adaptive FIR filter (where *A(n, q) = 0*). They have similar adaptive algorithms with similar convergence properties; the convergence rate and stability of the coefficient updates are usually determined by the eigenvalues of the Hessian matrix [3.5]. The main difference is that the equation-error adaptive IIR filter can operate as a pole-zero model by copying and inverting the polynomial *1 - A(n, q)*. The adaptive FIR filter is strictly an all zero model since *A (n,q) = 0.*

Equation (3.3) can also be compactly written as the inner product

$$y_e(n) = \theta^T(n)\phi_e(n) \tag{3.6}$$

where the coefficient vector θ and the signal vector each have length *M* + *N* - 1 and are defined as

$$\theta(n) = [a_1(n),.....,a_{N-1}(n),b_0(n),.....,b_{M-1}(n)]^T \tag{3.7a}$$

$$\phi_e(n) = [d(n-1),....,d(n-N+1),x(n),....,x(n-M+1)]^T \tag{3.7b}$$

Observe that (3.6) has the form of a linear regression, which is commonly used in statistics [3.6], where θ corresponds to the estimated parameters and ϕ_e is the regression vector (containing the data). The regressor is clearly independent of the coefficients since the data *d(n)* and *x(n)* are not functions of *A(n,* q) or *B(n, q)*. Many of the techniques and algorithms used in parametric statistical inference can be used here to find the optimal set of parameters. Some examples of these estimation methods are maximum likelihood [3.7], maximum a posteriori [3.7], least squares [3.8], and mean-square error [3.5]. The RLS (recursive-least-squares) algorithm [3.8] is one approach that recursively minimizes a least-squares criterion; the LMS (least-mean-square) algorithm [3.5] is a recursive gradient-descent method that searches for the minimum of the MSEE.

Output-error formulation

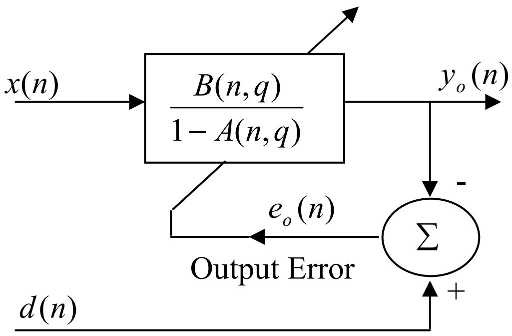


Fig. 3.4 Output -error formulation

The output-error adaptive IIR filter shown in Fig. 3.4 is implemented in direct form and is characterized by the following recursive difference equation (the subscript o denotes the output-error approach):

$$y_o(n) = \sum_{m=1}^{N-1} a_m(n) y_o(n-m) + \sum_{m=0}^{M-1} b_m(n) x(n-m) \tag{3.8}$$

which depends on past output samples $y_o(n - m)$, $m = 1, \ldots, (N-1)$. This output feedback significantly influences the form of the adaptive algorithm, adding greater complexity compared to that of the equation-error approach. Analogous to (3.4) and (3.6), this expression can be rewritten as the filter

$$y_o(n) = \left(\frac{B(n,q)}{1-A(n,q)} \right) x(n) \tag{3.9}$$

and as the inner product

$$y_o(n) = \theta^T(n) \phi_o(n) \tag{3.10}$$

where the coefficient vector θ is given in (3.7a) and the signal vector in this case is

$$\phi_o(n) = [y_o(n-1), \ldots, y_o(n-N+1), x(n), \ldots, x(n-M+1)]^T \tag{3.11}$$

The output $y_o(n)$ is clearly a nonlinear function of θ because the delayed output signals $y_o(n - k)$ of ϕ_o depend on previous coefficient values [i.e., they depend on $A(n - k, q)$ and

$B(n - k, q)$] . Equation (3.10) is not a linear regression, but it has the same form as (3.6) and is often referred to as a pseudolinear regression [3.9]. Similar statistical techniques and algorithms can be applied here to solve for the optimal coefficients, but it can be shown that the solution may be suboptimal unless a certain transfer function is strictly positive real (SPR) [3.10]. To overcome this SPR condition, additional processing (filtering) of the regression vector or of the output error is generally necessary, as will be shown later. The output error is given by $e_o(n) = d(n) - y_o(n)$, and it is called this simply because it is generated by subtracting the output in (3.9) from $d(n)$. Clearly, $e_o(n)$ is also a nonlinear function of θ; the mean-square-output error (MSOE) is, therefore, not a quadratic function and it can have multiple local minima [3.11]. Adaptive algorithms that are based on gradient-search methods could converge to one of these local solutions, resulting in suboptimal performance and inaccurate estimates of θ_*.

Clearly, there is a trade-off between the two error formulations. On the one hand, the equation error is a linear function of the coefficients so that the MSEE surface has only a global minimum and no local minima. The adaptive algorithms generally have fast convergence, but they converge to a biased solution since there is always some additive noise. On the other hand, the output error is a nonlinear function of the coefficients and the MSOE surface can have multiple local minima. The corresponding adaptive algorithms usually converge more slowly and they may converge to a local minimum. One might argue, however, that the output-error formulation is the “correct” approach because the adaptive filter is operating only on $x(n)$ such that the output $y(n)$ estimates the desired response $d(n)$. In contrast, the equation-error approach uses past values of the desired response as well as $x(n)$ to estimate $d(n)$.

3.3 Different Adaptive Algorithms

3.3.1 Deterministic Algorithm

A deterministic algorithm is an algorithm which, in informal terms, behaves predictably. Given a particular input, it will always produce the same correct output, and the underlying machine will always pass through the same sequence of states. Deterministic algorithms are by far the most studied and familiar kind of algorithm, as well as one of the most practical, since they can be run on real machines efficiently.

One simple model for deterministic algorithms is the mathematical; just as a function always produces the same output given a certain input, so do deterministic algorithms. The difference is that algorithms describe precisely how the output is obtained from the input, whereas abstract functions may be defined implicitly.

An adaptive algorithm is a procedure for adjusting the parameters of an adaptive filter to minimize a cost function chosen for the task at hand. In this section, we describe the general form of many adaptive IIR filtering algorithms and present a simple derivation of the LMS adaptive algorithm. In our discussion, we only consider an adaptive IIR filter structure, such that the output signal *y(n)* is given by (2.5). Such systems are currently more popular than adaptive FIR filters because (1) it contains both poles and zeros in the transfer function, while an FIR filter has only zeros, and (2) to achieve a specified level of performance, an IIR filter requires considerably fewer coefficients than the corresponding FIR filter.

3.3.1.1 General Form of Adaptive IIR Algorithms

The general structure of an adaptive IIR filter is shown below

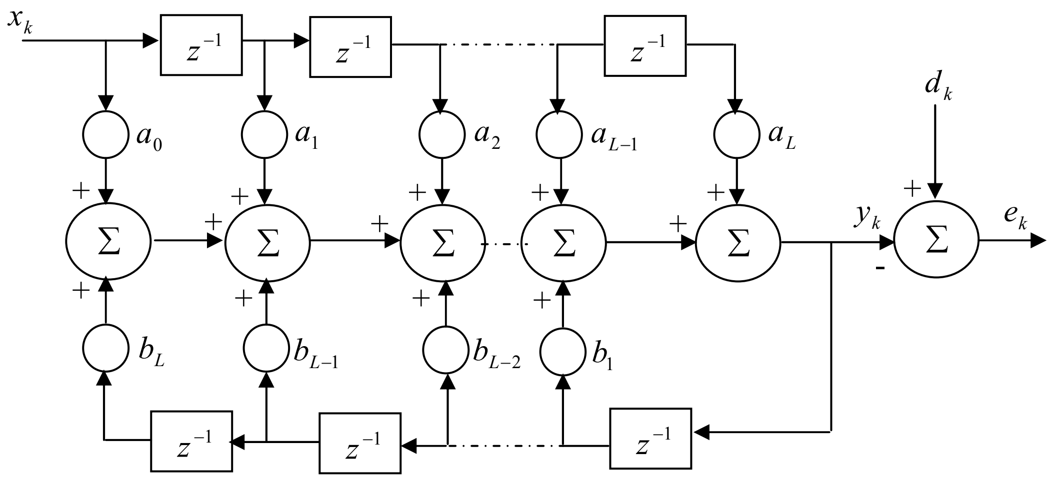


Fig 3.5 Single input digital IIR filter

The general form of an adaptive IIR filtering algorithm is

$$W(n+1) = W(n) + \mu(n)G(e(n), X(n)) \tag{3.12}$$

where G (-) is a particular vector-valued function, μ(n) is a *step size* parameter, *e(n)* and *X(n)* are the error signal and input signal vector, respectively, and the only information needed to adjust the coefficients at time *n* are the error signal, input signal vector, and step size.

The step size is so called because it determines the magnitude of the change or "step" that is taken by the algorithm in iteratively determining a useful coefficient vector. Much research effort has been spent characterizing the role that $\mu(n)$ plays in the performance of adaptive filters in terms of the statistical or frequency characteristics of the input and desired response signals. Often, success or failure of an adaptive filtering application depends on how the value of $\mu(n)$ is chosen or calculated to obtain the best performance from the adaptive filter.

The Mean-Squared Error Cost Function

The form of G(-) in (3.12) depends on the cost function chosen for the given adaptive filtering task. We now consider one particular cost function that yields a popular adaptive algorithm. Define the *mean-squared error* (MSE) cost function as

$$J_{MSE}(n) = \frac{1}{2}\int_{-\infty}^{\infty} e^2(n)\,p_n(e(n))\,de(n) \tag{3.13}$$

$$= \frac{1}{2}E\{e^2(n)\} \tag{3.14}$$

where $p_n(e)$ represents the probability density function of the error at time n and $E\{-\}$ is shorthand for the expectation integral on the right-hand side of (3.14). The MSE cost function is useful for adaptive IIR filters because

- $J_{MSE}(N)$ has a well-defined minimum with respect to the parameters in W(n);
- the coefficient values obtained at this minimum are the ones that minimize the power in the error signal *e{n),* indicating that *y(n)* has approached *d{n);* and
- J_{MSE} is a smooth function of each of the parameters in *W(n),* such that it is differentiable with respect to each of the parameters in *W(n).*

The third point is important in that it enables us to determine both the optimum coefficient values given knowledge of the statistics *of d(n)* and x(w) as well as a simple iterative procedure for adjusting the parameters of an IIR filter.

The Wiener Solution

For the FIR filter structure, the coefficient values in *W(n)* that minimize J_{MSE} (n) are well-defined if the statistics of the input and desired response signals are known. The formulation of this problem for continuous-time signals and the resulting solution was first derived by Wiener [3.4]. Hence, this optimum coefficient vector $W_{MSE}(n)$ is often called the Wiener solution to the adaptive filtering problem. The extension of Wiener's analysis to the discrete-time case is attributed to Levinson [3.12]. To determine $W_{MSE}(N)$ we note that the function $J_{MSE}(N)$ in (3.14) is quadratic in the parameters $\{w_i(n)\}$, and the function is also differentiable. Thus, we can use a result from optimization theory that states that the derivatives of a smooth cost function with respect to each of the parameters is zero at a minimizing point on the cost function error surface. Thus, $W_{MSE}(n)$ can be found from the solution to the system of equations

$$\frac{\partial J_{MSE}(n)}{\partial w_i(n)} = 0\,,\ 0 \le i \le L-1 \tag{3.15}$$

Taking derivatives of $J_{MSE}(N)$ in (3.14) and noting that $e(n)$ and $y(n)$ are given by the error and output signal, respectively, we obtain

$$\frac{\partial J_{MSE}(n)}{\partial w_i(n)} = E\{e(n)\frac{\partial e(n)}{\partial w_i(n)}\} \tag{3.16}$$

$$= -E\{e(n)\frac{\partial y(n)}{\partial w_i(n)}\} \tag{3.17}$$

$$= -E\{e(n)x(n-i)\} \tag{3.18}$$

$$= -(\,\mathrm{E}\{\mathrm{d(n)x(n-i)}\} - \sum_{\mathrm{j=0}}^{\mathrm{L-1}} \mathrm{E}\{\mathrm{x(n-i)x(n-j)}\}\mathrm{w_j}(n)) \tag{3.19}$$

where we have used the definitions of $e(n)$ and of $y(n)$ for the FIR filter structure and respectively, to expand the last result in (3.19). By defining the matrix $R_{xx}(n)$ and vector $P_{dx}(n)$ as

$$R_{XX} = E\{X(n)X^T(n)\} \text{ and } P_{dx}(n) = E\{d(n).X(n)\} \tag{3.20}$$

respectively, we can combine (3.15) and (3.19) to obtain the system of equations in vector form as

$$R_{XX}(n)W_{MSE}(n) - P_{dx}(n) = 0 \tag{3.21}$$

where 0 is the zero vector. Thus, so long as the matrix Rxx(n) is invertible, the optimum Wiener solution vector for this problem is

$$W_{MSE}(n) = R_{XX}{}^{-1}(n)P_{dx}(n) \tag{3.22}$$

The Method of Steepest Descent

The method of steepest descent is a celebrated optimization procedure for minimizing the value of a cost function *J(n)* with respect to a set of adjustable parameters W(n). This procedure adjusts each parameter of the system according to

$$w_i(n+1) = w_i(n) - \mu(n)\frac{\partial J(n)}{\partial w_i(n)} \tag{3.23}$$

In other words, the i^{th} parameter of the system is altered according to the derivative of the cost function with respect to the i^{th} parameter. Collecting these equations in vector form, we have

$$W(n+1) = W(n) - \mu(n)\frac{\partial J(n)}{\partial W(n)} \tag{3.24}$$

where $\partial J(n)/\partial W(n)$ is a vector of derivatives $dJ(n)/dw_i(n)$.

For an FIR adaptive filter that minimizes the MSE cost function, we can use the result in (3.19) to explicitly give the form of the steepest descent procedure in this problem. Substituting these results into (3.23) yields the update equation for W(n) as

$$W(n+1) = W(n) + \mu(n)(P_{dx}(n) - R_{XX}(n)W(n)) \tag{3.25}$$

However, this steepest descent procedure depends on the statistical quantities $E\{d(n)x(n-i)\}$ and $E\{x(n-i)x(n-j)\}$ contained in $P_{dx}(n)$ and $R_{xx}(n)$, respectively. In practice, we only have measurements of both *d(n)* and *x(n)* to be used within the adaptation procedure. While suitable estimates of the statistical quantities needed for (3.25) could be determined from the signals *x(n)* and *d{n),* we instead develop an approximate version of the method of steepest descent that depends on the signal values themselves. This procedure is known as the LMS algorithm.

3.1.1.2 The LMS Algorithm

The cost function *J(n)* chosen for the steepest descent algorithm of (3.23) determines the coefficient solution obtained by the adaptive filter. If the MSE cost function in (3.14) is chosen, the resulting algorithm depends on the statistics of *x(n)* and *d(n)* because of the expectation operation that defines this cost function. Since we typically only have measurements of *d(n)* and of *x(n)* available to us, we substitute an alternative cost function that depends only on these measurements. One such cost function is the least squares cost function given by

$$J_{LS}(n) = \sum_{k=0}^{n} \alpha(k)(d(k) - W^T(n)X(k))^2 \tag{3.26}$$

where $\alpha(n)$ is a suitable weighting sequence for the terms within the summation. This cost function, however, is complicated by the fact that it requires numerous computations to calculate its value as well as its derivatives with respect to each *W(n),* although efficient recursive methods for its minimization can be developed. Alternatively, we can propose the simplified cost function $J_{LMS}(N)$ given by

$$J_{LMS}(n) = \frac{1}{2}e^2(n) \tag{3.27}$$

This cost function can be thought of as an instantaneous estimate of the MSE cost function, as $J_{MSE}(n) = E\{J_{LMS}(n)\}$. Although it might not appear to be useful, the resulting algorithm obtained when $J_{LMS}(N)$ is used for *J(n)* in (3.23) is extremely useful for practical applications. Taking derivatives of $J_{LMS}(N)$ with respect to the elements of W(n) and substituting the result into (3.23), we obtain the LMS adaptive algorithm given by

$$W(n+1) = W(n) + \mu(n)e(n)X(n) \tag{3.28}$$

Note that this algorithm is of the general form in (3.12). It also requires only multiplications and additions to implement. In fact, the number and type of operations needed for the LMS algorithm is nearly the same as that of the FIR filter structure with fixed coefficient values, which is one of the reasons for the algorithm's popularity.

The behavior of the LMS algorithm has been widely studied, and numerous results concerning its adaptation characteristics under different situations have been developed. For now, we indicate its useful behavior by noting that the solution obtained by the LMS algorithm near its convergent point is related to the Wiener solution. In fact, analyses of the LMS algorithm under certain statistical assumptions about the input and desired response signals show that

$$\lim_{x \to \infty} E\{W(n)\} = W_{MSE} \tag{3.29}$$

when the Wiener solution $W_{MSE}(n)$ is a fixed vector. Moreover, the average behavior of the LMS algorithm is quite similar to that of the steepest descent algorithm in (3.25) that depends explicitly on the statistics of the input and desired response signals. In effect, the iterative nature of the LMS coefficient updates is a form of time-averaging that smoothes the errors in the instantaneous gradient calculations to obtain a more reasonable estimate of the true gradient.

The problem is that gradient descent is a local optimization technique, which is limited because it is unable to converge to the global optimum on a multimodal error surface if the algorithm is not initialized in the basin of attraction of the global optimum.

Several modifications exist for gradient-based algorithms in attempt to enable them to overcome local optima. One approach is to simply add noise or a momentum term to the gradient computation of the gradient descent algorithm to enable it to be more likely to escape from a local minimum. This approach is only likely to be successful when the error surface is relatively smooth with minor local minima, or some information can be inferred about the topology of the surface such that the additional gradient parameters can be assigned accordingly. Other approaches attempt to transform the error surface to eliminate or diminish the presence of local minima [3.13], which would ideally result in a unimodal error surface. The problem with these approaches is that the resulting minimum transformed error used to update the adaptive filter can be biased from the true minimum output error and the algorithm may not be able to converge to the desired minimum error condition. These algorithms also tend to be complex, slow to converge, and may not be guaranteed to emerge from a local minimum. Some work has been done with regard to removing the bias of equation error LMS [3.13] and Steiglitz-McBride [3.14] adaptive IIR filters, which add further complexity with varying degrees of success.

Recursive LMS Algorithm

Fig (3.5) represents the general single-input linear combiner or digital filter. Without the feedback portion, the filter is called”nonrecursive”. With the feedback the portion is called “recursive” .The expression for output, $y_{k,}$ in this case is

$$y_k = \sum_{n=0}^{L} a_n x_{k-n} + \sum_{n=1}^{L} b_n y_{k-n} \tag{3.30}$$

To develop algorithms for the recursive adaptive filter, let us place the recursive filter in fig 3.6 in the standard adaptive model, as illustrated below:

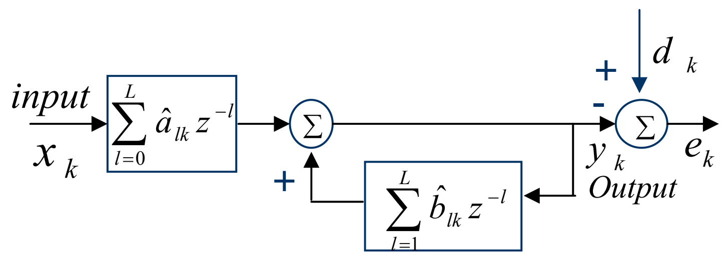


Fig3.6 Recursive adaptive filter

Here the vector X_k can represent either the multiple-or single-input situation, and of course y_k is a scalar in equation (3.30).This form applies to the single-input case, which we assume her for convenience. A time-varying weight vector and a new signal vector, W_k and U_k, are now defined as given below.

An IIR filter [3.15] involves both feed forward and feed back path .The presence of feedback means that portions of the filter output and possible other internal variables in the filter are feedback the input. Consequently, unless the filter is properly designed, feedback can make it unstable, with the result that the filter oscillates; this kind of operation is clearly unacceptable when the requirement is that stability is "must". The structure of a direct-form IIR filter is shown in Fig. 3.5. In this case, the output of the system can be mathematically represented as

$$y(n) = \sum_{i=1}^{N} a_i y(n-i) + \sum_{j=0}^{N} b_j(n) x(n-j) \tag{3.31}$$

$$W_k = [a_{0k}\ a_{1k} \ldots\ldots a_{Lk} b_{1k} \ldots\ldots b_{Lk}]^T$$

$$U_k = [x_k\ x_{k-1} \ldots\ldots x_{k-L}\, y_{k-1} \ldots\ldots y_{k-L}]^T$$

From Fig. 3.6, we can write

$$\begin{aligned} e_k &= d_k - y_k \\ &= d_k - W_k^T U_k \end{aligned}$$

Now, considering the gradient based LMS algorithm in Eq. (3.16)

$$\hat{\nabla}_k = \frac{\partial e_k^2}{\partial W_k}$$
$$= 2e_k \left[\frac{\partial e_k}{\partial a_{0k}} ...... \frac{\partial e_k}{\partial a_{Lk}} \frac{\partial e_k}{\partial b_{1k}} ...... \frac{\partial e_k}{\partial b_{Lk}} \right]^T \tag{3.32}$$
$$= -2e_k \left[\frac{\partial y_k}{\partial a_{0k}} ...... \frac{\partial y_k}{\partial a_{Lk}} \frac{\partial y_k}{\partial b_{1k}} ...... \frac{\partial y_k}{\partial b_{Lk}} \right]^T$$

Using Eq. (3.32), we define

$$\alpha_{nk} = \frac{\partial y_k}{\partial a_n} = x_{k-n} + \sum_{l=1}^{L} b_l \frac{\partial y_{k-l}}{\partial a_n}$$
$$= x_{k-n} + \sum_{l=1}^{L} b_l \alpha_{n,k-l} \tag{3.33}$$
$$\beta_{nk} = \frac{\partial y_k}{\partial b_n} = y_{k-n} + \sum_{l=1}^{L} b_l \frac{\partial y_{k-l}}{\partial b_n}$$
$$= y_{k-n} + \sum_{l=1}^{L} b_l \beta_{n,k-l} \tag{3.34}$$

With the derivatives defined in this manner, we have

$$\hat{\nabla}_k = -2e_k \left[\alpha_{0k} \quad \dots \quad \alpha_{Lk} \beta_{1k} \quad \dots \quad \beta_{Lk} \right] \tag{3.35}$$

$$\mu(k) = diag[\mu_0 .... \mu_L] \tag{3.36}$$

Now, we can summarize the LMS algorithm for recursive filter as follows:

$$\begin{aligned}
y_k &= W_k^T U_k \\
\alpha_{nk} &= x_{k-n} + \sum_{l=1}^{L} b_l \alpha_{n,k-l} \\
\beta_{nk} &= y_{k-n} + \sum_{l=1}^{L} b_l \beta_{n,k-l} \\
\hat{\nabla}_k &= -2e_k \left[\alpha_{0k} \quad \dots \quad \alpha_{Lk} \beta_{1k} \quad \dots \quad \beta_{Lk} \right] \\
W_{k+1} &= W_k - \mu(k) \hat{\nabla}_k
\end{aligned}$$

Table 3.1: LMS algorithm for recursive filter

Thus Table (3.1) describes recursive LMS. The calculation of gradient factors contains forward co-efficients as well as backward co-efficients. The gain constant is problem specific, and usually chosen by trail and error method.

These types of recursive adaptive filter have following disadvantages:

1. Instability may come into picture if the poles move outside the unit circle during the adaptation.
2. Performance surface are generally nonquadratic and may even have local minima.

These are serious disadvantages, and consequently the recursive adaptive filters has had very limited applications. Hence an entire class of algorithms known as the HARF (hyperstable adaptive recursive filter) algorithms has been proposed by Larimore at al [3.16].

The simplest member of this class is SHARF (Simplified HARF) [3.16].In this algorithm we have used a smoothed version of e_k , obtained by filtering e_k. This is described in Eq. (3.37).

$$\begin{aligned}
y_k &= W_k^T U_k \\
e_k &= d_k - y_k \\
v_k &= e_k + \sum_{n=1}^{N} c_n e_{k-n} \\
\hat{\nabla}_k &= -2 v_k \left[x_k \quad \ldots \quad x_{k-L}\, y_{k-1} \quad \ldots \quad y_{k-L} \right]^T \\
W_{k+1} &= W_k - \mu(k) \hat{\nabla}_k
\end{aligned} \tag{3.37}$$

The e's value used in Eq. (3.37) keeps a major role for convergence. This algorithm avoids use of gradient rule; hence it avoids local minima as well as multimodal error surface. This SHARF algorithm [3.16] is easy to be implemented because of its stability.

Another approach, attempts to locate the global optimum by running several LMS algorithms in parallel, initialized with different initial coefficients. The notion is that a larger, concurrent sampling of the error surface will increase the likelihood that one process will be initialized in the global optimum valley. This technique does have potential, but it is inefficient and may still suffer the fate of a standard gradient technique in that it will be unable to locate the global optimum if none of the initial estimates is located in the basin of attraction of the global optimum. By using a similar congregational scheme, but one in which information is collectively exchanged between estimates and intelligent randomization is introduced, structured stochastic algorithms are able to hill-climb out of local minima. This enables the algorithms to achieve better, more consistent results using a fewer number of total estimates. These types of algorithms provide the framework for the algorithms discussed in the following sections.

3.1.1.3 Recursive Least Square (RLS) Algorithms

The algorithms that result from the above approach have the disadvantages that they can be slow to approach the optimal weight vector and, once close to it, usually"rattle around" the optimal; vector rather than actually converge to it, due to the effects of approximations made in the estimate of the performance function gradient. To overcome these difficulties, another approach is discussed in this section. Here we develop algorithms that use the input data *{x,d}* in such a way as to ensure optimality at each step. If we can be done, then clearly the result of the algorithm for the last data point is the overall optimal weight vector.

Suppose that we refine the sum squared performance function J_{ss} by the expression

$$J_k = \sum_{l=N-1}^{k-1} \left| y(l) - d(l) \right|^2 , \ N-1 \le k \le L-1 \tag{3.38}$$

This form of J simply reflects how much data have been used so far. Clearly, J_L uses all the available data from $k=0$ to $k=L-1$. Suppose we define W_k^o as the impulse response vector that minimizes J_k .By this definition, W_{L-1}^o equals W_{ss}^o, and the optimal impulse vector over all the data.

The motivation for developing "recursive-in-time" algorithms can be seen as follows. Suppose *x(l)* and *d(l)* have been received for time up through k-1 and that W_k^o has been computed. Now suppose that *x(k)*and *d(k)*are received, allowing us to form

$$J_{k+1} = \sum_{l=N-1}^{k} \left| y(l) - d(l) \right|^2 \stackrel{\Delta}{=} J_k + \left| y(k) - d(k) \right|^2 \tag{3.39}$$

We desire to find some procedure by which W_k^o can be updated to produce W_{k+1}^o**,** the new optimal vector. If we can develop such a procedure, then we can build up the optimal weight vector step by step until the final pair of data points *x(L-1)* are received. With these points, W_{L-1}^o can be computed, which, by definition, is the global optimum vector W_{ss}^o.

The update formula:

The simplest approach to updating W_k^o is the following procedure:

(a)Update R_{ss}	$R_{ss,k+1} = R_{ss,k} + X(k)X^t(k)$
(b) Update P_{ss}	$P_{ss,k+1} = P_{ss,k} + d(k)X(k)$
(c) Invert	$R_{ss,k+1}$
(d)Compute W^o_{k+1}	$W^o_{k+1} = R^{-1}_{ss,k+1} P_{ss,k+1}$

Table 3.2: Update equation for RLS algorithm

The autocorrelation matrix and crosscorrelation vectors are updated and then used to compute W^o_{k+1}. While direct, this technique is computationally wasteful. Approximately N^3+2N^2+N multiplications is required at each update, where N is the impulse response length, and have that N^3 are required for the matrix inversion if done with the classical Gaussian elimination technique.

In an effort to reduce the computational requirement for this algorithm, we focus first on this inversion. We notice that Gaussian elimination makes no use whatsoever have the special form of $R_{ss,k}$ or of the special form of the update from $R_{ss,k}$ to $R_{ss,k+1}$. We now set out to take advantage of it. We do so by employing the well-known matrix inversion lemma, also sometimes called the *ABCD* lemma,

$$(A+BCD)^{-1} = A^{-1} - A^{-1}B(DA^{-1}B + C^{-1})^{-1} DA^{-1} \tag{3.40}$$

We use this lemma by making the following associations:

$$\begin{aligned} A &= R_k \\ B &= X(k) \\ C &= 1 \\ D &= X^t(k) \end{aligned} \tag{3.41}$$

With these associations, R_{k+1} can be represented as

$$R_{k+1} = R_k + X(k)X^t(k) = A + BCD \tag{3.42}$$

and R^{-1}_{k+1} is given by

$$R^{-1}_{k+1} = R^{-1}_k - \frac{R^{-1}_k X(k) X^t(k) R^{-1}_k}{1 + X^t(k) R^{-1}_k X(k)} \tag{3.43}$$

Thus, given R_k^{-1} and a new input *x(k),* hence *X(k)* **,**we can compute R_{k+1}^{-1} directly. We never compute R_{k+1}, nor do we invert it directly.

The optimal weight vector W_{k+1}^{o} is given by

$$W_{k+1}^{o} = R_{k+1}^{-1} P_{K+1} \tag{3.44}$$

which can be obtained by combining (3.43) with update P_{ss}

$$W_{k+1}^{o} = \left\{ R_k^{-1} - \frac{R_k^{-1} X(k) X^t(k) R_k^{-1}}{1 + X^t(k) R_k^{-1} X(k)} \right\} . \left\{ P_k + d(k) X(k) \right\}$$

$$= R_k^{-1} P_k - \frac{R_k^{-1} X(k) X^t(k) R_k^{-1} P_k}{1 + X^t(k) R_k^{-1} X(k)} + d(k) R_k^{-1} X(k) - \frac{d(k) . R_k^{-1} X(k) X^t(k) R_k^{-1} X(k)}{1 + X^t(k) R_k^{-1} X(k)} \tag{3.45}$$

To simplify this result, we make the following associations and definitions. The k^th optimal weight vector:

$$R_k^{-1} P_k = W_k^{o} \tag{3.46}$$

The filtered information vector:

$$Z_k \stackrel{\Delta}{=} R_k^{-1} X(k) \tag{3.47}$$

The priori output:

$$y_o(k) \stackrel{\Delta}{=} X^t(k) W_k^{o} \tag{3.48}$$

The normalized input power:

$$q = X^t(k) Z_k = X^t(k) R_k^{-1} X(k) \tag{3.49}$$

With these expressions, the optimal weight vector W_{k+1}^{o} becomes

$$\begin{aligned} W_{k+1}^{o} &= W_k^{o} - \frac{Z_k X^t(k) W_k^{o}}{1 + X^t(k) Z_k} + d(k) Z_k - \frac{d(k) Z_k X^t(k) Z_k}{1 + X^t(k) Z_k} \\ &= W_k^{o} - \frac{Z_k y_o(k)}{1+q} + d(k) Z(k) - \frac{d(k) q Z_k}{1+q} \\ &= W_k^{o} - \frac{Z_k y_o(k)}{1+q} + \frac{d(k) Z_k}{1+q} \\ &= W_k^{o} + \frac{\left\{ d(k) - y_o(k) \right\} . Z_k}{1+q} \end{aligned} \tag{3.50}$$

Equations (3.43) and (3.46)-(3.50) comprise the *recursive least squares (RLS)* algorithm.

Steps for RLS Algorithm:

The step-by-step procedures for updating W_k^o are given in this section. This set of steps is efficient in the sense that no unneeded variable is computed and that no needed variable is computed twice. We do, however, need assurance that R_k^{-1} exists. The procedure then goes as follows:

(i) Accept new samples *x(k), d(k)*.

(ii) Form *X(k)* by shifting *x(k)* into the information vector.

(iii) Compute the *a priori* output $y_o(k)$:

$$y_o(k) = W_k^{ot} x(k) \tag{3.51}$$

(iv) Compute *a priori* error $e_o(k)$:

$$e_o(k) = d(k) - y_o(k) \tag{3.52}$$

(v) Compute the filtered information vector Z_k :

$$Z_k = R_k^{-1} X(k) \tag{3.53}$$

(vi) Compute the normalized error power q:

$$q = X^t(k) Z_k \tag{3.54}$$

(vii) Compute the gain constant *v:*

$$v = \frac{1}{1+q} \tag{3.55}$$

(viii) Compute the normalized filtered information vector $\widetilde{Z}_k$:

$$\widetilde{Z}_k = v.Z_k \tag{3.56}$$

(ix) Update the optimal weight vector W_k^o to W_{k+1}^o**:**

$$W_{k+1}^o = W_k^o + e_o(k)\widetilde{Z}_k \tag{3.57}$$

(x) Update the inversion correlation matrix R_k^{-1} to R_{k+1}^{-1} in preparation for the next iteration:

$$R_{k+1}^{-1} = R_k^{-1} - \widetilde{Z}_k \widetilde{Z}_k^t \tag{3.58}$$

This procedure assumes that R_k^{-1} exists at the initial time in the recursion. As a result, two initialization procedures are commonly used. The first is to build up R_k and Pk until R has full rank, i.e. at least N input vectors X(k) are acquired. At this point R_k^{-1} is computed directly and then W_k. Given these, the recursion can proceed as described above indefinitely or until $k=L-1$. The advantage of the first technique is that optimality is preserved at each step. The major price paid is that is about N^3 computations are required once to perform that initial inversion.

A second, much simpler approach is also commonly used. In this case R_{N-1}^{-1} is initialized as

$$\hat{R}_{n-1}^{-1} = \eta I_N \tag{3.59}$$

where η is a large positive constant and I_N is the N-by-N identity matrix. Since R_{N-1}^{-1} almost certainly will not equal ηI_N, this inaccuracy will influence the final estimate of R_k and hence W_k. A s a practical matter, however, η can usually be made large enough to avoid significant impact on W_{L-1}^{o} while still making R_{N-1} invertible. Because of the simplicity and the low computational cost, the second approach is the one of the most commonly used. It becomes even more theoretically justifiable when used with the exponentially weighted RLS algorithm to be discussed shortly.

The computational cost for the RLS algorithm:

As a prelude to developing even more efficient adaptive algorithms, we first should determine how much computation is required to execute the RLS algorithm.

We define that the 10 steps in the procedure can be grouped by their computational complexity:

(a) Order 1: Steps (iv) and (vii) require only a few simple operations, such as a subtraction or an addition and division. These are termed as *order1* and denoted *O(1)* because the amount of computation required is not related to the filter order.

(b) Order N: Steps (iii), (vi), (viii), and (ix) each require a vector dot product, a scalar-vector product, or a vector scale and sum operation. Each of these requires N additions for each iteration of the algorithm .The actual number of multiplications required for these steps is $4N$, but we refer to them as order N, or $O(N)$,because the computation requirement is proportional to N, the length of the filter impulse response.

(c) Order N^2: Step (v), a matrix vector product, and step (x), the vector outer product, both require N^2 multiplications and approximately N^2 additions. These are termed $O(N^2)$ procedures.

The total number of computations needed to execute the RLS algorithm for each input sample pair { $x(k), d(k)$ } is $2N^2+4N$ multiplications, an approximately equal number of additions ,and on division. Because this amount of computation is required for each sample pair, the total requirement of multiplications to process the sample window is

$$C_{RLS} = (L-N+1).\ 2N^2 + (L-N+1).\ 4N$$

There are several reasons for exploring and using RLS techniques:

(a) RLS can be numerically better behaved than the direct inversion of R_{ss};

(b) RLS provides an optimal weight vector estimate at every sample time, while the direct method produce a weight vector estimate only at the end of the data sequence; and

(c) This recursive formulation leads the way to even lower-cost techniques.

3.3.2 Structured Stochastic Approaches

Stochastic search algorithms aim at increasing the probability of encountering the global minimum, without performing an exhaustive search of the entire parameter space. Whereas the previously described approaches (and gradient approaches in general) rely on the adaptive filter structure to update the filter parameters, a structure independent global optimization approach is a stochastic search of the error space. In structure independent optimization, a gradient is not calculated and the structure of the adaptive filter does not directly influence the parameter updates - aside from the error computation. Due to this property, these types of algorithms are capable of globally optimizing any class of adaptive filter structures or objective functions by assigning the parameter estimates to represent filter tap weights or any other possible parameter of the unknown system model - even the exponents of polynomial terms.

3.3.2.1. Genetic Algorithm

Genetic algorithms are a part of evolutionary computing, which is a rapidly growing area. Genetic algorithms are inspired by Darwin's theory of evolution [3.17]. Simply said, problems are solved by an evolutionary process resulting in a best (fittest) solution (survivor) - in other words, the solution is evolved.

Genetic Algorithms (GA) are based upon the process of natural selection and does not require error gradient statistics. As a consequence, a GA is able to find a global error minimum [3.18]. The acceptance of GA optimization across many fields has been slow due to the lack of a mathematical derivation. Published results have, however, demonstrated the advantage of the GA optimization and have aided in changing this perception in many disciplines ([3.19],[3.20],[3.21],[3.22],[3.23],[3.24]).

GA's have been applied to many applications that have previously used ineffective and unstable optimization techniques. The IIR filter is one such example. The IIR error surface is known to be multimodal, gradient-learning algorithms become either unstable or stuck within local minima [3.25]. 'Evolutionary' approaches have been applied to the adaptive IIR filter to overcome these learning problems ([3.26]).

The Algorithm begins with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are then selected to form new solutions (offspring) are selected according to their fitness - the more suitable they are the more chances they have to reproduce. This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied.

Outline of the Basic Genetic Algorithm

The outline of the Basic GA is very general. There are many parameters and settings that can be implemented differently in various problems.

The first question to ask is how to create chromosomes and what type of encoding to choose. We then address Crossover and Mutation; the two basic operators of GA encoding, crossover and mutation are introduced next.

1. **[Start]** Generate random population of *n* chromosomes (suitable solutions for the problem)
2. **[Fitness]** Evaluate the fitness *f(x)* of each chromosome *x* in the population
3. **[New population]** Create a new population by repeating following steps until the new population is complete
    1. **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
    2. **[Crossover]** With a crossover probability cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.
    3. **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).
    4. **[Accepting]** Place new offspring in the new population
4. **[Replace]** Use new generated population for a further run of the algorithm
5. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in current population
6. **[Loop]** Go to step **2**

Table 3.3: Outline of the Basic Genetic Algorithm

The next question is how to select parents for crossover. This can be done in many ways, but the main idea is to select the better parents (best survivors) in the hope that the better parents will produce better offspring.

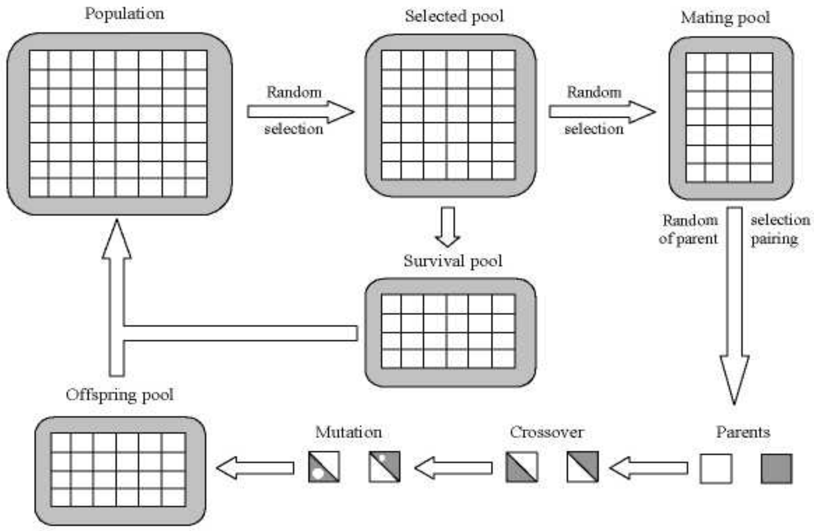


Fig 3.7 A GA iteration cycle.

From the population a pool of individuals is randomly selected, some of these survive into the next iterations population. A mating pool is randomly created and each individual is paired off. These pairs undergo evolutionary operators to produce two new individuals that are added to the new population.

Population Variables

A chromosome consists of the problem variables, where these can be arranged in a vector or a matrix. In the gene crossover process, corresponding genes are crossed so that there is no inter-variable crossing and therefore each chromosome uses the same fixed structure. An initial population that contains a diverse gene pool offers a better picture of the cost surface where each chromosome within the population is initialized independently by the same random process.

In the case of binary-genes each bit is generated randomly and the resulting bit-words are decoded into their real value equivalent The binary number is used in the genetic search process and the real value is used in the problem evaluation. This type of initialization results in a normally distributed population of variables across a specific range. This type of results in a normally distributed population of variables across a specific range.

A GA population, *P*, consists of a set of *N* chromosomes *{Cj... C_N}* and *N* fitness values $\{f_1 \ldots\ldots f_N\}$, where the fitness is some function of the error matrix.

$$P = \{(C_1, f_1)\,(C_2, f_2)\,(C_3, f_3)\cdots(C_N, f_N)\} \tag{3.60}$$

The GA is an iterative update algorithm and each chromosome requires its fitness to be evaluated individually. Therefore, N separate solutions need to be assessed upon the same training set in each training iteration. This is a large evaluation overhead where population sizes can range between twenty and a hundred, but the GA is seen to have learning rates that evens this overhead out over the training convergence.

Chromosome Selection

The selection process is used to weed out the weaker chromosomes from the population so that the more productive genes may be used in the production of the next generation. The chromosome finesses are used to rank the population with each individual assigned it own fitness value, *f.*

$$E_i(n) = \frac{1}{M}\sum_{j=1}^{M} e_{ji}^{\,2}(n) \tag{3.61}$$

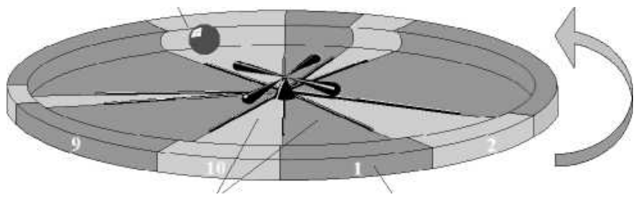
The solution cost value E_i of the chromosome in the population is calculated from a training-block of M training signals in Eq. (3.61) and from this cost an associated fitness is assigned:

$$f_i(n) = \frac{1}{(1+E_i(n))} \tag{3.62}$$

The fitness can be considered to be the inverse of the cost but the fitness function in Eq (3.62) is preferred for stability reasons, i.e. $E_i(n) = 0$.When the fitness of each chromosome in the population has been evaluated, two pools are generated, a survival pool and a mating pool. The chromosomes from the mating pool will be used to create a new set of chromosomes through the evolutional processes of natural selection and the survival pool allows a number of chromosomes to pass onto the next generation. The chromosomes are selected randomly for the two pools but biased towards the fittest. Each chromosome may be chosen more than once and the fittest chromosomes are more likely to be chosen so that they will have a greater influence in the new generation of solutions.

The selection procedure can be described using a biased roulette wheel with the buckets of the wheel sized according to the individual fitness relative to the population's total fitness [3.18]. Consider an example population often chromosomes that have the fitness assessment of f = {0.16, 0.16, 0.48, 0.08, 0.16, 0.24, 0.32, 0.08, 0.24, 0.16} and the sum of the finesses are used to normalize these values, *fmm*=2.08.

Figure 3.10 shows a roulette wheel that has been split into ten segments and each segment is in proportion to the population chromosomes relative fitness. The third individual has the highest fitness and nearly accounts for a quarter of the total fitness. The third segment therefore fills nearly a quarter of the roulette wheels area. The random selector points to a chosen chromosome, which is then copied into the mating pool because the third individual controls a greater proportion of the wheel, it has a greater probability of being selected.



Chromosome segments *Population roulette wheel*

Fig3.10 Biased roulette-wheel that is used in the selection of the mating pool.

The commonly used reproduction operator is the proportionate reproductive operator where a string is selected from the mating pool with a probability proportional to the fitness. Thus ith string in the population is selected with a probability proportional to f_i where f_i is the fitness value of that string.Since the population size is usually kept fixed in a simple GA,the sum of probabilities of each string being selected for the mating pool must be one. The probability of ith-selected string is

$$p_i = \frac{f_i}{\sum_{j=1}^{n} f_j} \tag{3.63}$$

Where 'n' is the population size.

Using the fitness value f_i of all strings, the probability of selecting a string p_i can be calculated. There after, cumulative probability P_i of each string can be calculated by adding the individual probabilities from the top of the list. Thus the bottom most string in the population should have a cumulative probability of 1.The roulette wheel concept can be simulated by realizing that the ith string in the population represents the cumulative probability from P_{i-1} to P_i. Thus the first string represents the cumulative values from 0 to P_1. Hence cumulative probability of any string lies between 0-1. In order to choose n strings, n random numbers between zero and one is created at random. Thus the string that represents the chosen random number in the cumulative probability range(calculate from fitness value) for the string, is copied from to the mating pool. This way the string with a higher fitness value will represent a larger range in the cumulative probability values and therefore, has a higher probability of being copied into the mating pool. On the other hand string with a smaller fitness value will represent a smaller range in the cumulative probability values and therefore, has a lesser probability of being copied into the mating pool.

After the GA crossover and mutation operators update the selected mating pool chromosomes, these supercede the old population and consequently the genes from the unselected chromosomes are lost.

Gene Crossover

The crossover operator exchanges gene information between two selected chromosomes, (C_q, C_r), where this operation aims to improve the diversity of the solution vectors. The pair of chromosomes, taken from the mating pool, becomes the parents of two offspring chromosomes for the new generation.

In the case of a binary crossover operation the least significant bits are exchanged between corresponding genes within the two parents. For each gene-crossover a random position along the bit sequence is chosen and then all of the bits rights of the crossover point are exchanged. In Fig.3.11 (a), which shows a single point crossover, the fifth crossover position is randomly chosen, where the first position corresponds to the left side. The bits from the right of the fourth bit will be exchanged. Fig.3.11 (b) shows a two-point crossover in which two points are randomly chosen and the bits in between them are exchanged.

1 0 1 0 ¦ 0 1 0 1	Before crossover
0 0 1 0 ¦ 1 1 1 0	
1 0 1 0 ¦ 1 1 1 0	After crossover
0 0 1 0 ¦ 0 1 0 1	

Fig 3.11(a) Single point crossover

1 ¦ 0 1 0 0 1 ¦ 0 1	Before crossover
0 ¦ 0 1 0 1 1 ¦ 1 0	
1 ¦ 0 1 0 1 1 ¦ 0 1	After crossover
0 ¦ 0 1 0 0 1 ¦ 1 0	

Fig 3.11(b) Double point crossover

Fig3.11 shows a basic genetic crossover with the same crossover point chosen for both offspring genes. At the start of the learning process the extent of crossing over the whole population can be decided allowing the evolutionary process to randomly select the individual genes. The probability of a gene crossing, *P(crossing)*, provides a percentage estimate of the genes that will be affected within each parent. *P(crossing)*≤1 allows all the gene values to be crossed and *P(crossing)*=0 leaves the parents unchanged, where a random gene selection value, ω Є {1,0}, is governed by this probability of crossing.

The crossover does not have to be limited to this simple operation. The crossover operator can be applied to each chromosome independently, taking different random crossing points in each gene. This operation would be more like grafting parts of the original genes onto each other to create the new gene pair. All of a chromosome's genes are not altered within a single crossover. A probability of gene-crossover is used to randomly select a percentage of the genes and those genes that are not crossed remain the same as one of the parents.

Chromosome Mutation

The last operator within the breeding process is mutation. Each chromosome is considered for mutation with a probability that some of its genes will be mutated after the crossover operation.

Selected bit for mutation

1	0	1	1	$\boxed{0}$	0	1	0	Before mutation
1	0	1	1	$\boxed{1}$	0	1	0	After mutation

Fig.3.12 Mutation operation in GA

A random number is generated for each gene, if this value is within the specified mutation selection probability, *P (mutation)*, the gene will be mutated. The probability of mutation occurring tends to be low with around one percent of the population genes being affected in a single generation. In the case of a binary mutation operator, the state of the randomly selected gene-bits is changed, from zero to one or vice-versa.

A simple genetic algorithm treats the mutation as a secondary operator with the role of restoring lost genetic materials. Suppose, for example, all the string in a population have conveyed to to a zero at a given position and the optimal solution has a one at that position, then crossover cannot regenerate a one at that position while a mutation could. It helps the search algorithm to escape from local minima's traps since the modification is not related to any previous genetic structure of the population. The mutation is also used to maintain diversity in the population .For example, consider the following population having four eight-bit strings.

0110 1011
0011 1101
0001 0110
0111 1100

All the four strings have a zero in the left most bit position. If the true optimum solution requires a one in that position, then neither reproduction nor crossover operator will be able to create a one in that position. Only mutation operation can change that zero to one.

Parameters of GA

I. Crossover and Mutation Probability

There are two basic parameters of GA - crossover probability and mutation probability.

Crossover probability: This probability controls the frequency at which the crossover occurs for every chromosome in the search process. This is a number between (0,l) which is determined according to the sensitivity of the variables of the search process. The crossover probability is chosen small for systems with sensitive variables. If there is crossover, offspring are made from parts of both parent's chromosome. Crossover is made in hope that new chromosomes will contain good parts of old chromosomes and therefore the new chromosomes will be better. However, it is good to leave some part of old population survives to next generation.

Mutation probability: This parameter decides how often parts of chromosome will be mutated. If there is no mutation, offspring are generated immediately after crossover (or directly copied) without any change. If mutation is performed, one or more parts of a chromosome are changed. If mutation probability is 100%, whole chromosome is changed, if it is 0%, nothing is changed. Mutation generally prevents the GA from falling into local extremes. Mutation should not occur very often, because then GA will in fact change to random search.

II. Other Parameters

There are also some other parameters in GA. One important parameter is population size.

Population size: How many chromosomes are in population in one generation. If there are too few chromosomes, GA has few possibilities to perform crossover and only a small part of search space is explored. On the other hand, if there are too many chromosomes, GA slows down. Research shows that after some limit (which depends mainly on encoding and the problem) it is not useful to use very large populations because it does not solve the problem faster than moderate sized populations.

3.4. Results and Discussion

In this section, we study the response matching of the desired and estimated signal. And also study the convergence performance using computer simulations for different order filters. The input signal x(n) is a white random process with a uniform distribution and unit variance with additive noise ,was also a white random process uncorrelated with x(n) unit. An estimate of the MSE at each instant of time was obtained by averaging $|e(n)|^2$ over 50 independent computer runs.

3.4.1 LMS Based Adaptive IIR System Identification

The input signal *x(n)* is passes through the plant having transfer function *H(z)* which then added with white Guassian noise of –30dB.The plant is nothing but an IIR filter having transfer function of different orders which are consider below. In this discussion, we consider in account up to 5th order filter. The step size for the entire order filter was taken is same i.e.0.25 and the number of samples taken is 3000.

Case 1: The plant is second order (M=2; L=2). It has two forward path coefficients and two feed back path coefficients. The transfer function of the plant is given by:

$$H(z) = \frac{1.25z^{-1} - 0.25z^{-2}}{1 - 0.3z^{-1} + 0.4z^{-2}}$$

From the fig.3.13, it was shown that the response matching between the desired and estimated signal is not good enough and the normalized mean square error reduces to –26dB.

After adaptation, the coefficients of the adaptive processor are approximately equal to that of the plant. The NMSE is converged approximately 2200 number of iterations.

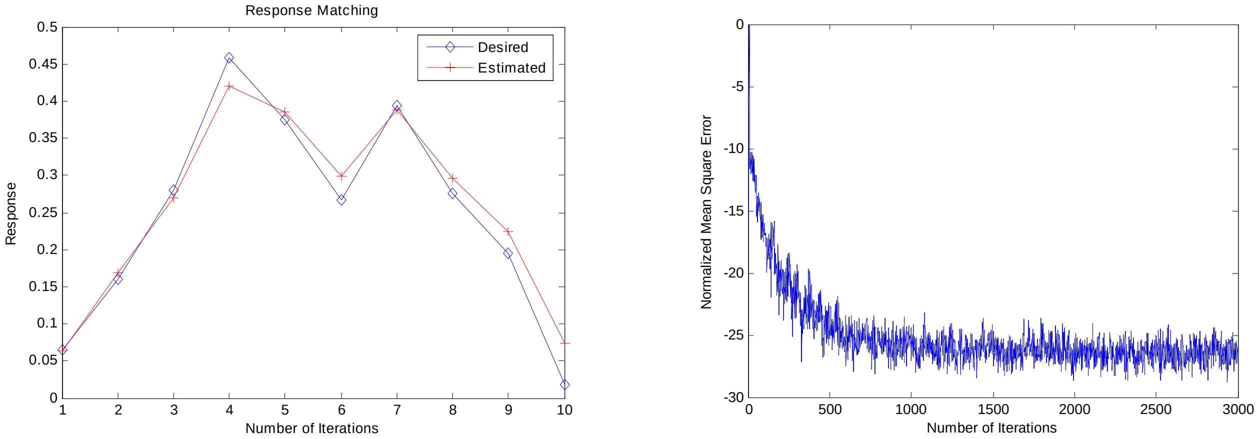


Fig.3.13. Performance study of 2nd order LMS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

Case 2: Here the plant is assumed to be 3rd order IIR filter as given below:

$$H(z) = \frac{0.6z^{-1} - 0.25z^{-2} + 0.2z^{-3}}{1 - 0.2z^{-1} - 0.4z^{-2} + 0.5z^{-3}}$$

For the response matching between desired and estimated signal, 3000 numbers of iterations are taken here. But in order to see more clearly, only 10 samples are taken into consideration. In this, the normalized mean square error reduces to –29dB.The NMSE floor converge approximately 2000 number of iterations.

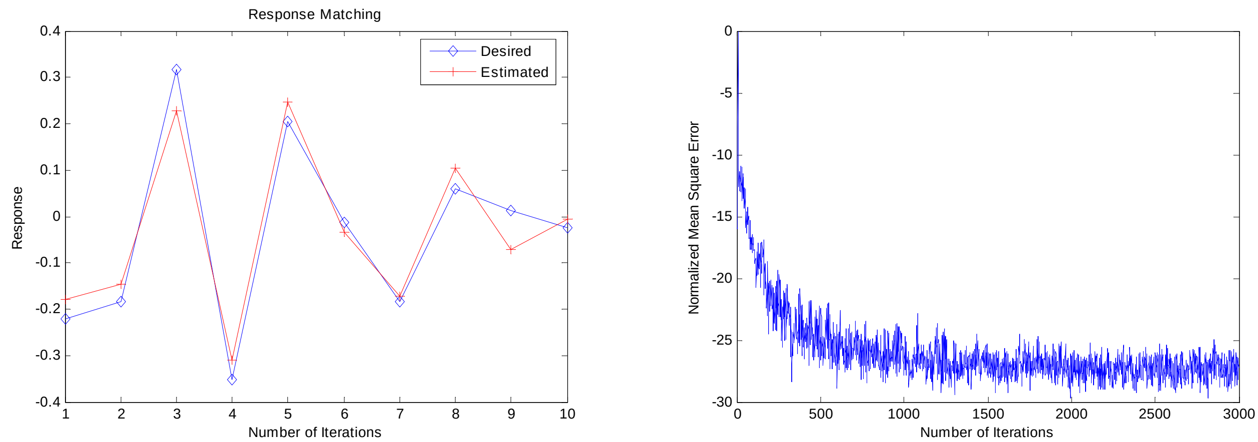


Fig.3.14. Performance study of 3rd order LMS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

Case 3: The transfer function of the plant is assumed to be 4th order as shown below:

$$H(z) = \frac{1 - 0.9z^{-1} + 0.81z^{-2} - 0.729z^{-3}}{1 - 0.23z^{-1} + 0.433z^{-2} - 0.339z^{-3} + 0.518z^{-4}}$$

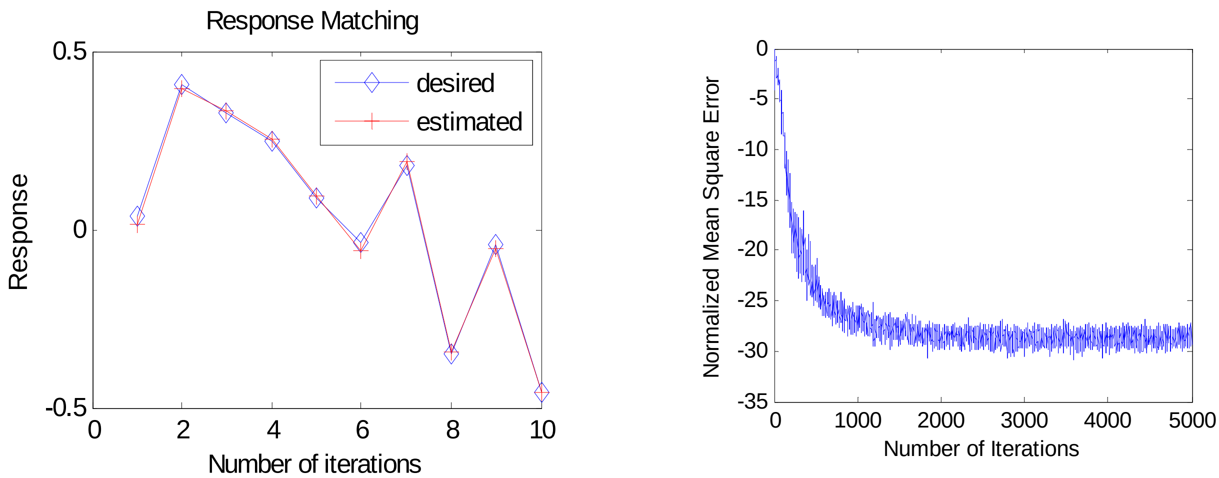


Fig.3.15 Performance study of 4th order LMS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

From the above figure, it was shown that the response matching between the desired and estimated signal is good and the normalized mean square error reduces to –30dB.With extensive computer simulation, it was shown that the transfer function of the adaptive processor is approximately equal to that of the plant. The NMSE converges approximately after 1800 number of iterations.

Case 4: The transfer function of the adaptive IIR filter is assumed here is 5th order as shown below:

$$H(z) = \frac{0.1084 + 0.5419z^{-1} + 1.0837z^{-2} + 1.0837z^{-4} + 0.1084z^{-5}}{1 + 0.9853z^{-1} + 0.9738z^{-2} + 0.3864z^{-3} + 0.1112z^{-4} + 0.0113z^{-5}}$$

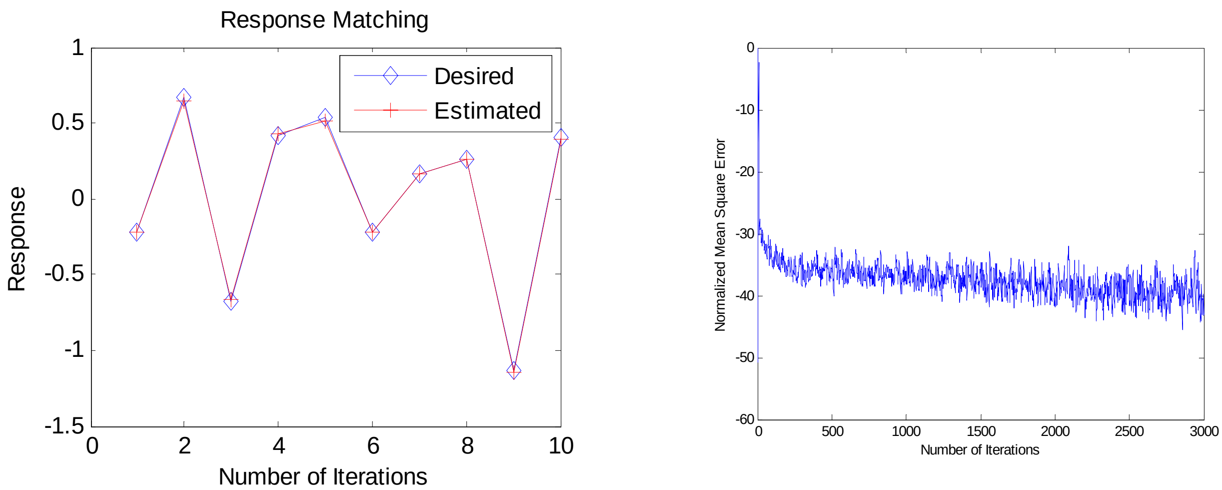


Fig.3.16 Performance study of 5th order LMS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

From the above simulation study, it was shown that the response matching between the desired and estimated signal is superior and the normalized mean square error floor reaches –40dB.After adaptation, the coefficients of the adaptive processor are close to that of the plant. Hence as the order of the filter increases, the response matching between the desired and the estimated signal gives superior performance and the performance convergence of MSE floor is better.

3.4.2 RLS Based Adaptive IIR System Identification

For the computer simulation, -30dB noise is added to the output of the plant and 3000 numbers of iterations are taken into account. δ ,which is a very large number for adaptation in RLS algorithm is assumed to be 0.0025 for all the cases.

Case 1: The plant is second order (M=2; L=2). It has two forward path coefficients and two feed back path coefficients. The transfer function of the plant is given by:

$$H(z) = \frac{0.05 - 0.4z^{-1}}{1 - 1.1314z^{-1} + 0.25z^{-2}}$$

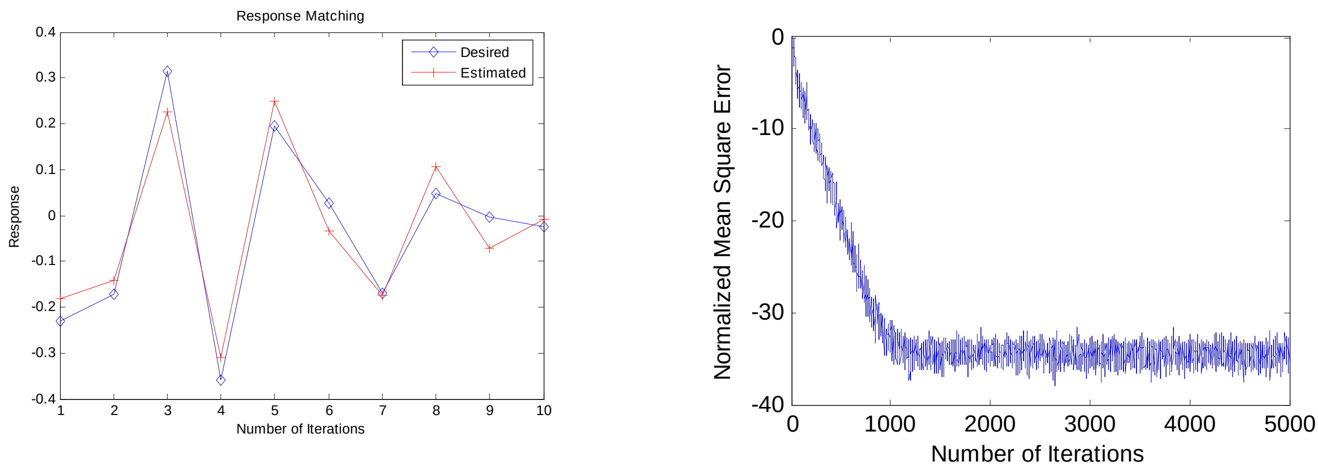


Fig.3.17 Performance study of 2nd order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

From the above fig it was shown that the NMSE floor reaches –33dB for approximately 1100 number of iterations. For more convenience, 10 numbers of samples are shown for the testing part.

Case 2: Here the plant is assumed to be 3rd order IIR filter as given below:

$$H(z) = \frac{1.2 - 0.5z^{-1} + 0.1z^{-2}}{1 - 0.3z^{-1} + 0.4z^{-2} + 0.5z^{-3}}$$

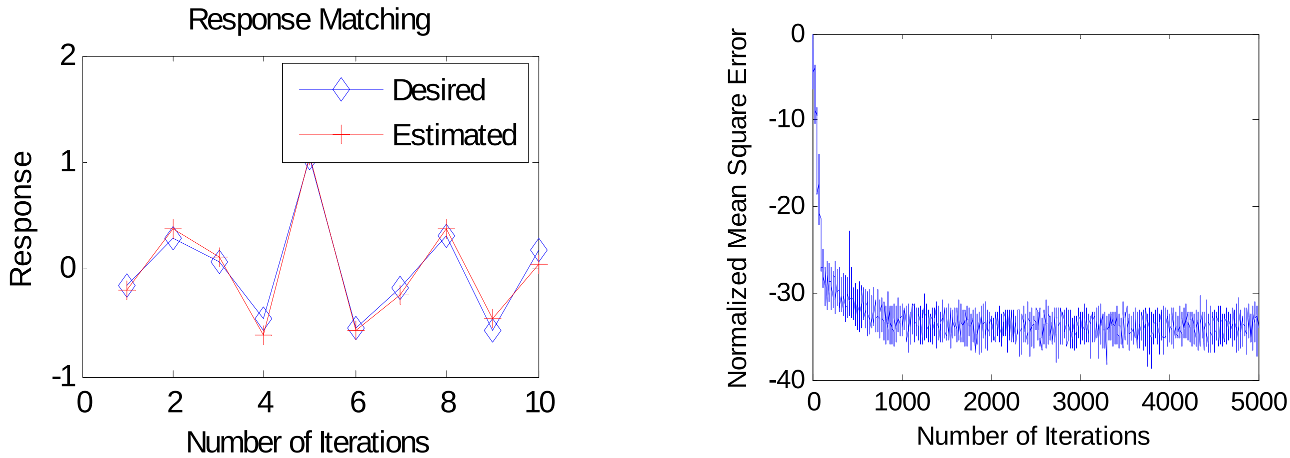


Fig.3.18 Performance study of 3rd order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

Case 3: The transfer function of the plant is assumed to be 4th order as shown below:

$$H(z)=\frac{1-0.9z^{-1}+0.81z^{-2}-0.729z^{-3}}{1-0.23z^{-1}+0.433z^{-2}-0.339z^{-3}+0.518z^{-4}}$$

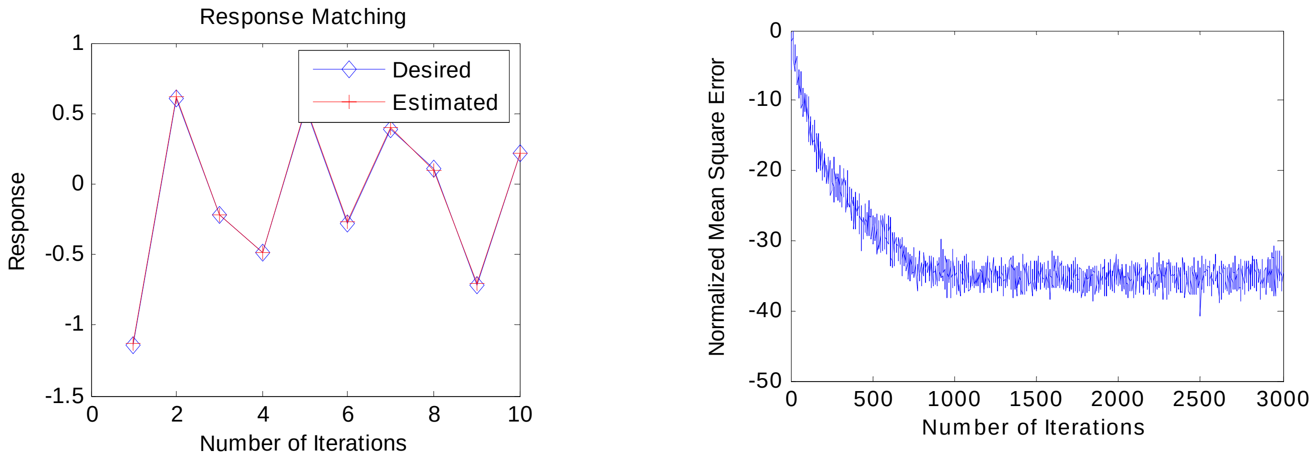


Fig.3.19 Performance study of 4th order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

Case 4: The transfer function of the adaptive IIR filter is assumed here is 5th order as shown below:

$$H(z)=\frac{0.1084+0.5419z^{-1}+1.0837z^{-2}+1.0837z^{-4}+0.1084z^{-5}}{1+0.9853z^{-1}+0.9738z^{-2}+0.3864z^{-3}+0.1112z^{-4}+0.0113z^{-5}}$$

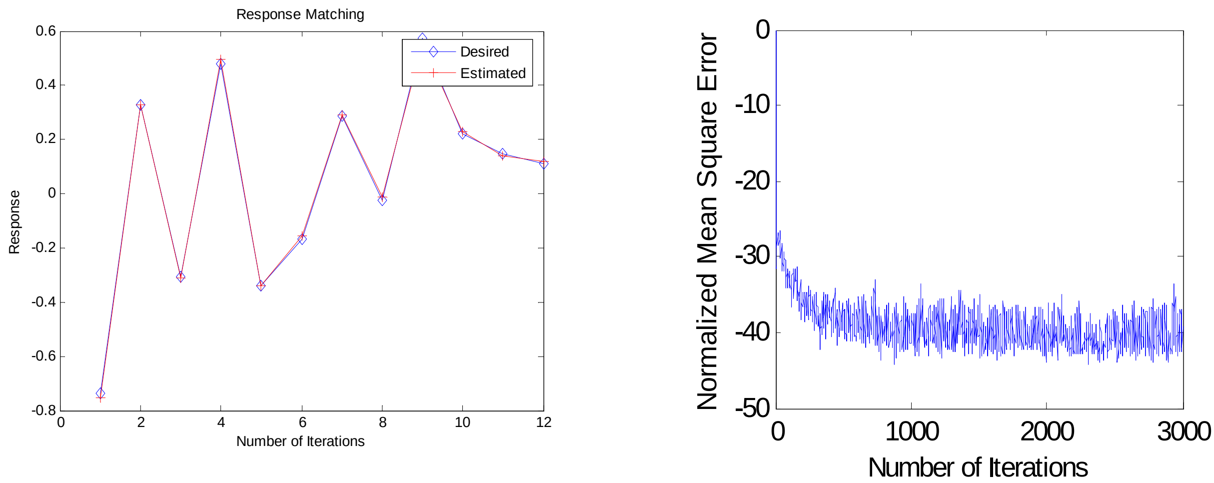


Fig.3.Performance study of 5th order RLS based adaptive IIR filter
(a) Response match between desired and estimated
(b) NMSE plot in dB

From all the simulation study of LMS and RLS based adaptive IIR system identification; it observed that the RLS algorithm gives better performance than the LMS convergence.

3.4.3 GA Based IIR system Identification

Case 1: The plant is assumed here is 2nd order as shown below:

$$H(z) = \frac{1.25z^{-1} - 0.25z^{-2}}{1 - 0.3z^{-1} + 0.4z^{-2}}$$

GA Parameters	BCGA
Population N	60
Crossover probability	0.8
Mutation probability	0.1
Number of generations cycles	50

Table 3.4: Simulation data for GA based 2nd order adaptive IIR filter

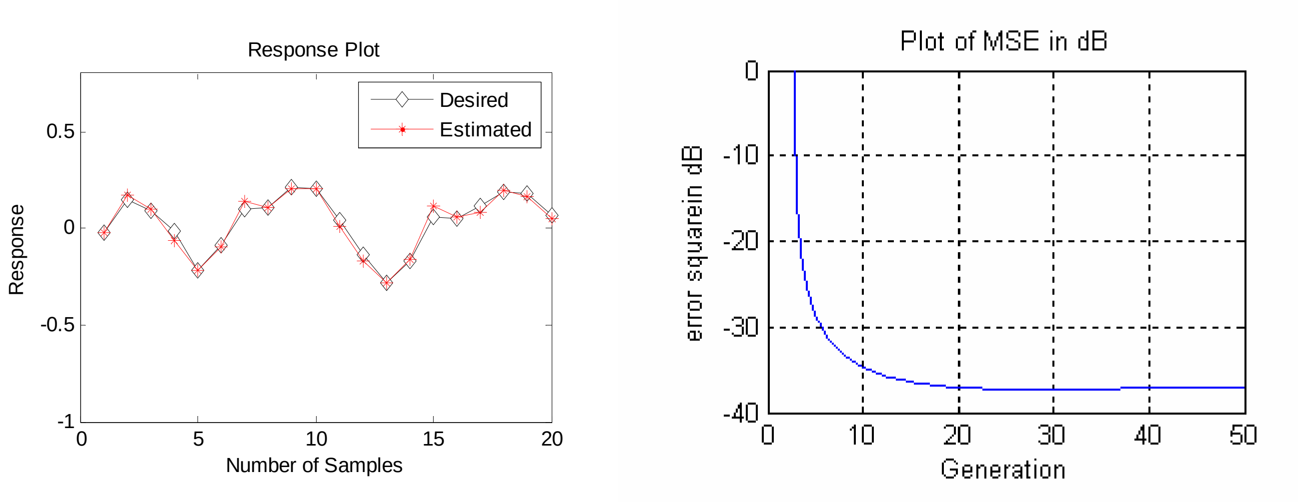


Fig.3.19 Performance study of GA based 2nd order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

Case 2: Here the plant is assumed to be 3rd order IIR filter as given below:

$$H(z) = \frac{1.2 - 0.5z^{-1} + 0.1z^{-2}}{1 - 0.3z^{-1} + 0.4z^{-2} + 0.5z^{-3}}$$

GA Parameters	BCGA
Population N	80
Crossover probability	0.85
Mutation probability	0.15
Number of generations cycles	50

Table 3.5: Simulation data for GA based 3rd order adaptive IIR filter

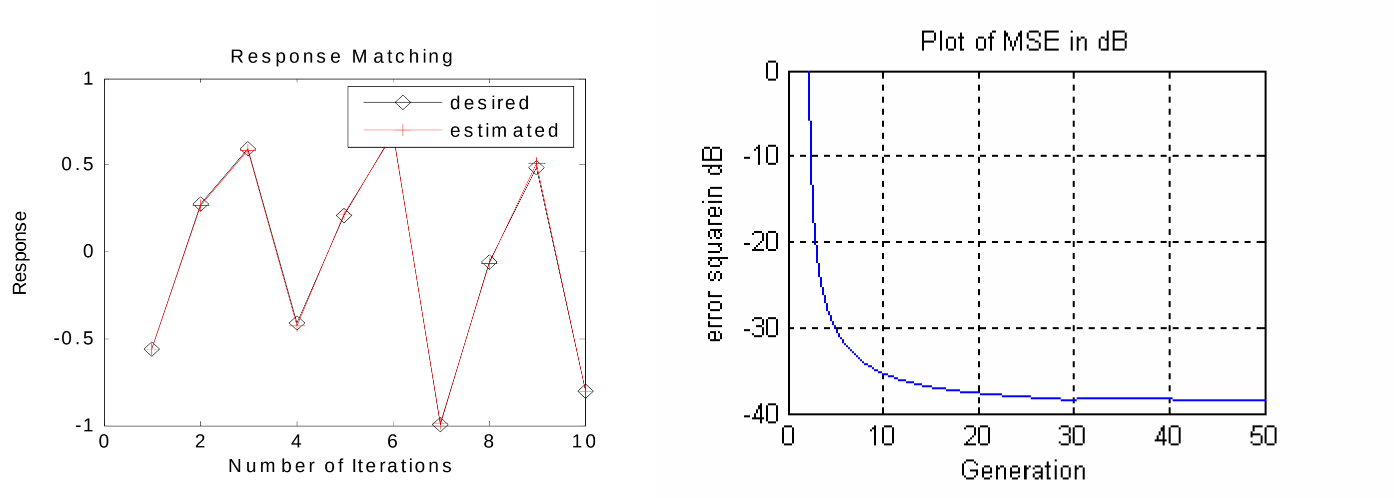


Fig.3.20. Performance study of GA based 3rd order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

Case 3: The transfer function of the plant is assumed to be 4th order as shown below:

$$H(z) = \frac{1 - 0.9z^{-1} + 0.81z^{-2} - 0.729z^{-3}}{1 - 0.23z^{-1} + 0.433z^{-2} - 0.339z^{-3} + 0.518z^{-4}}$$

GA Parameters	BCGA
Population N	110
Crossover probability	0.9
Mutation probability	0.1
Number of generations cycles	50

Table 3.6: Simulation data for GA based 4th order adaptive IIR filter

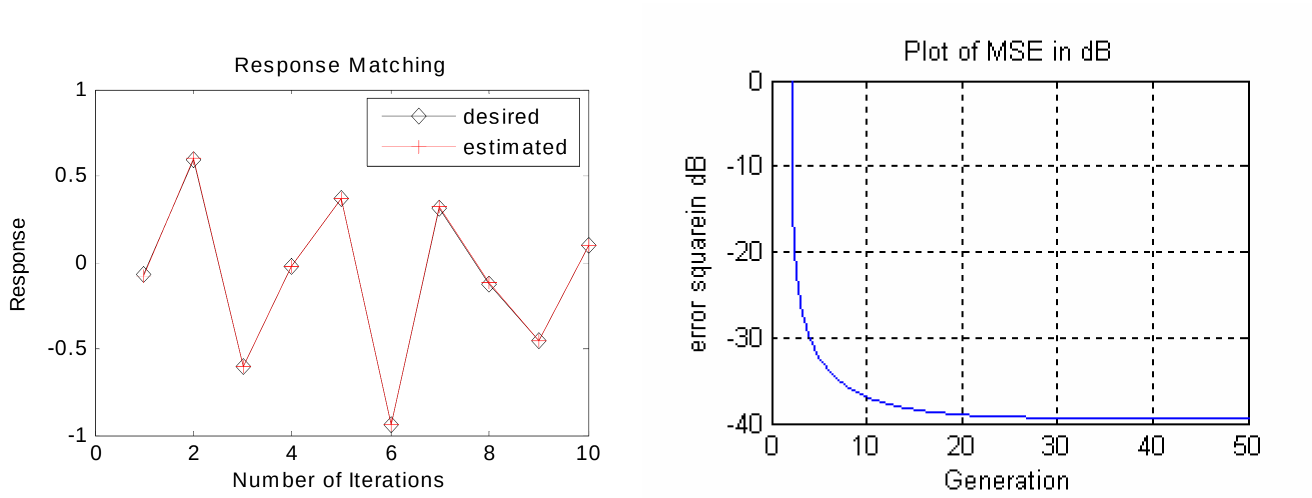


Fig.3.21 Performance study of GA based 4th order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

Case 4: The trasfer function of the adaptive IIR filter is assumed here is 5th order as shown below:

$$H(z) = \frac{0.1084 + 0.5419z^{-1} + 1.0837z^{-2} + 1.0837z^{-4} + 0.1084z^{-5}}{1 + 0.9853z^{-1} + 0.9738z^{-2} + 0.3864z^{-3} + 0.1112z^{-4} + 0.0113z^{-5}}$$

GA Parameters	BCGA
Population N	120
Crossover probability	0.8
Mutation probability	0.15
Number of generations cycles	50

Table 3.7: Simulation data for GA based 5th order adaptive IIR filter

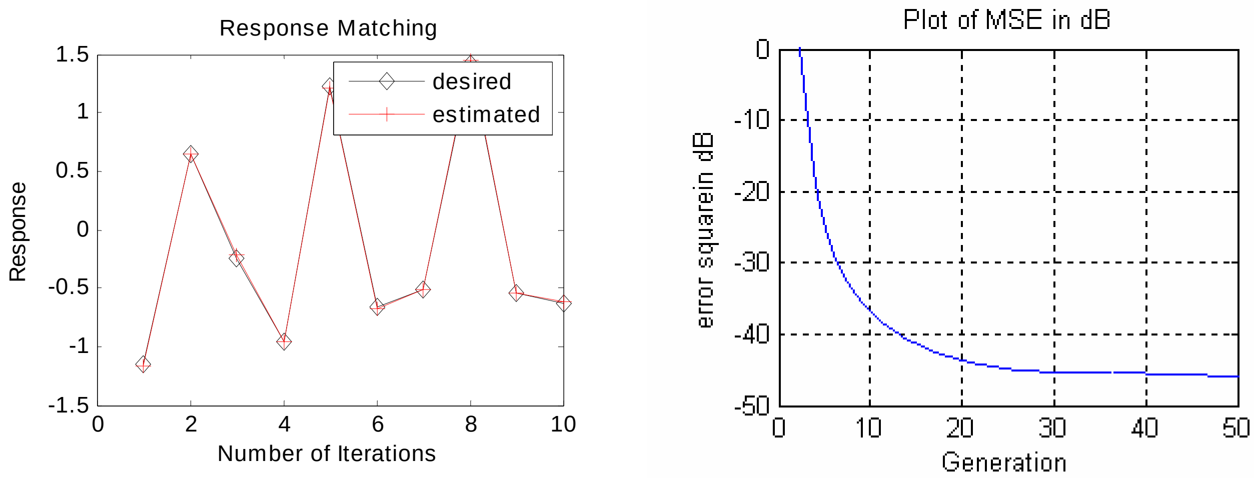


Fig.3.22 Performance study of GA based 5th order adaptive IIR filter
(a) Response match between desired and estimated
(b) MSE plot in dB

The convergence is greatly improved by using the genetic search algorithm. From all the computer simulation study, it is observed that from case1 to case 4 of GA based adaptive IIR filter, GA converges to the global solution with -37dB, - 39dB,-40dB, and -45dB squared error respectively.

Comparison of computational complexity for different algorithms

Algorithms(structure) for IIR Filter	Computational Complexity	Description
LMS	(L+M)(L+2)	L=backward length of IIR filter. M=forward length of IIR filter
RLS	$2N^2+4N$	N=L+M; L=backward length of IIR filter; M=forward length of IIR filter
GA	O (N.POPSIZE)	N=L+M; POPSIZE=no. of chromosomes in a population; M=no. of offsprings; POPSIZE>>m minimum calculations required in one generation=N.POPSIZE.

Table 3.8: Comparison of computational complexity for different algorithms

3.5 Summary

In this chapter LMS, RLS and GA were introduced and extensively explained. It was stated that the RLS based adaptive IIR system identification gives superior performance than the LMS counterpart. Because of the multimodal error surface of IIR filtering, special attention has to be given to the learning algorithms in order to avoid the local minimum problem. The new algorithm introduce here is the GA can be applied to both unimodal and multimodal search surfaces for optimization where in the later case gradient descent algorithms face difficulties. The experimental results clearly show that the GA based adaptive IIR system identification outperforms the LMS and RLS based adaptive IIR system identification in terms of the convergence speed and the ability to locate the global optimum solution.

*****

Chapter 4

ADAPTIVE IIR CHANNEL EQUALIZATION

Adaptive model for channel equalization
Nonminimum phase channel equalization
Results and discussion

CHAPTER 4

4. ADAPTIVE IIR CHANNEL EQUALIZATION

4.1 Introduction

In an ideal communication channel, the received information is identical to that transmitted. However, this is not the case for real communication channels, where signal distortions take place. A channel can interfere with the transmitted data through three types of distorting effects: power degradation and fades, multi-path time dispersions and background thermal noise [4.1]. Equalization is the process of recovering the data sequence from the corrupted channel samples. A typical baseband transmission system is depicted in Fig 4.1, where an equalizer is incorporated within the receiver [4.2].

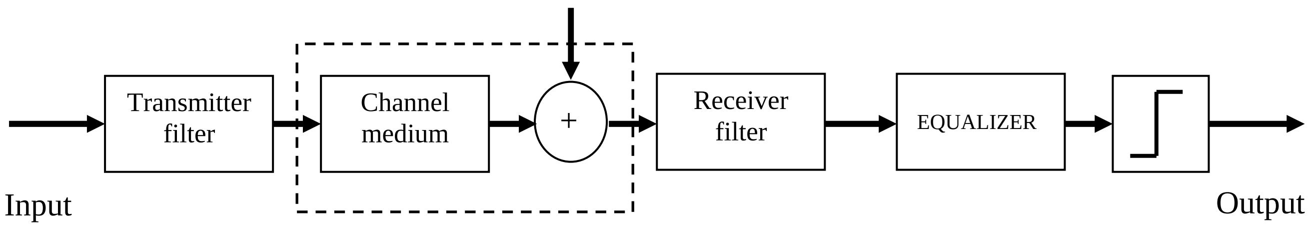


Fig 4.1 A baseband communication System

The equalization approaches investigated in this thesis are applied to a BPSK (binary phase shift keying) baseband communication system. Each of the transmitted data belongs to a binary and 180° out of phase alphabet {-1, +1} within this chapter channel baseband models are explained. A transversal equalizer structure is also examined, [4.3].

4.2 Adaptive model for channel equalization

Equalizers are among the most essential digital signal processing devices for digital communication systems. This technology has been developed since the mid-sixties and seventies, [Lucky 1965, Sondhi 1967, Qureshi 1985 and Widrow 1976], and the research in this areas is continuing to provide better solutions and performances. One of the reasons for this on going research is due to the ever increasing demands for higher capacity and efficient bandwidth utilization.

Communication channels such as telephone, wireless and optical channels are susceptible to intersymbol interference (ISI). Without channel equalization, the utilization of the channel bandwidth becomes inefficient. Channel equalization is a process of compensating for the effects caused by a band-limited channel, hence enabling higher data rates [Qureshi 1985]. These disruptive effects are due to the dispersive transmission medium (e.g. telephone cables) and the multipath effects in the radio channel. A typical communication system is depicted in Figure 3.1, where the equalizer is incorporated within the receiver while the channel introduces intersymbol interference. The transfer function of the equalizer is an estimate of the direct inverse of the channel transfer function. To transmit high-speed data over a band-limited channel, the frequency response of the channel is usually not known with sufficient precision to design an optimum match filter. The equalizer is, therefore, designed to be adaptive to the channel variation. The configuration of an adaptive equalizer is depicted in Figure 4.2. Based on the observed channel output, an adaptive algorithm recursively updates the equalizer to reconstruct the output signal.

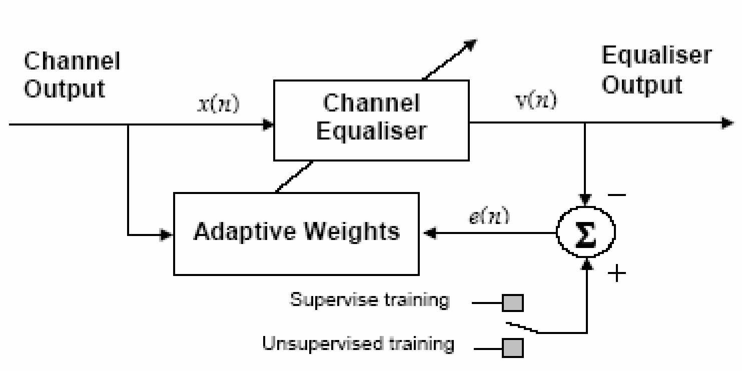


Figure 4.2. A simple channel equalizer configuration.

There are two modes of equalization, supervised and unsupervised training. Supervised training employs a training sequence from a pre-stored sequence inside the receiver or embedded in the transmitted sequence. Unsupervised training, also called decision-directed equalization, employs a decision device to return the noisy estimated symbols to the actual symbols to be used to train the equalizer.

The optimum equalization performance (BER) can be obtained by using a maximum likelihood sequence estimator (MLSE) on the transmitted data sequence [Forney 1972]. However, the optimum equalizer is computationally expensive to implement. Moreover, the complexity increases with longer channel length and with tracking a time-varying channel.

The sub-optimal equalizer is simple and more robust such as the linear transversal equalizer (LTE) or the decision feedback equalizer (DFE). The main disadvantage of the sub-optimal equalizer is that it takes a longer time to converge before a creditable data transmission can take place. Sub-optimal equalizers use an adaptive algorithm to update the transversal tap delay filter. Hence, in general, its simplicity outweighs the performance offered by the MLSE.

Adaptive equalization deconvolves the effects of a communication channel or some other system and produces an inverse model of an unknown plant. In Fig.4.3, the adaptive processor attempts to recover a delayed version of the signal ‘s’, which is assumed to have been altered by the slowly varying channel with additive noise. Instability occurs when poles are moved outside the unit circle at the time of adaptation for IIR systems.

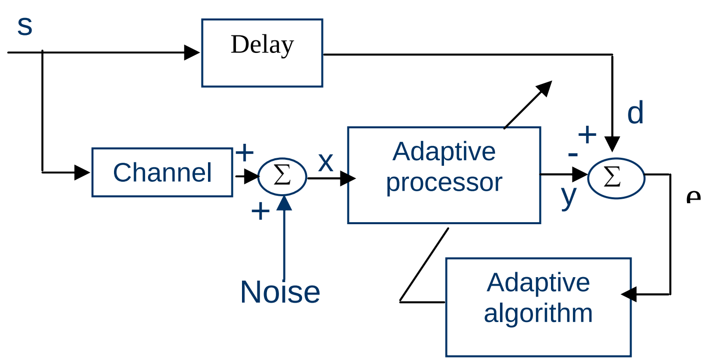


Fig.4.3. Channel equalization model

Basically there are two types of equalizer structure, linear and non-linear. Decision feedback equalizers and transversal equalizers are considered linear because the internal structure is a linear combiner. However, in order the study the gradient descent-based adaptive algorithms' performance, the linear equalizer is more appropriate.

Minimum and non-minimum phase channel:

Fundamentally there are two types of communication channels, the minimum phase and the non-minimum phase channel. A minimum phase channel can be identified when all the zeros of the z-transform channel model lie within the unit circle [Macchi 1995]. In contrast, the non-minimum phase channel is defied when there are zeroes outside the unit circle. Equations (3.1) and (3.2) show simple examples of the minimum and non-minimum phase channels respectively.

$$H(z)=1.0 + 0.5z^{-1}, \qquad (4.1)$$

$$H(z) = 0.5 + z^{-1}, \tag{4.2}$$

A direct inverse of the minimum phase channel in equation (3.1) is a stable convergent series; $(1.0 + 0.5z^{-1})^{-1} = \sum_{i=0}^{\infty}(-\frac{1}{2})^i z^{-i}$. For the non-minimum phase channel, its direct inverse is a divergent series, which is unstable; $(0.5 + z^{-1})^{-1} = z\sum_{i=0}^{\infty}(-\frac{1}{2})^i z^i$. For the channel equalization, only truncations of the most significant coefficients are required, because it is not practical to include all coefficients. For the non-minimum phase channel, a time delay is incorporated into the training signal to cope with the non-causality. Therefore, either in minimum phase or non-minimum phase channels, the equalizer can be used to equalize the channel distortion.

4.3 An adaptive IIR equalizer for nonminimum-phase channels

In communication systems, the multipath channel is a major obstacle in reliable communication. Numerous works have been done to solve the problem [4.4], [4.5]. The particular characteristics of the multipath channel are generally unknown. Thus, one of the objectives of the receiver is to accurately identify the channel characteristics in order to correct the channel effect. In most such channels adaptive equalization schemes are used.

Conventional equalizers fall into two categories. One is “inverse modeling” and the other is “system modeling”. The former is also referred to as equalizer configuration or direct equalization and the latter as system identification configuration or indirect equalization. The inverse modeling requires an FIR filter with a large number of taps to approximate the inverse of a channel but has an advantage of being always stable. The system modeling requires much less number of taps but may be unstable for nonminimum-phase channels that are common in the real world. So, if we can extract the minimum-phase part from an unknown channel, it is possible to optimally equalize the channel by using a hybrid configuration in which both direct and indirect equalizers are combined to form an IIR equalizer.

In this section, we present, an adaptive IIR equalizer based on extraction of the minimum-phase part, from an unknown channel. A direct equalizer (DE) and an equalized channel identifier (ECI) are adapted simultaneously for obtaining a minimum-phase equalized channel. Once the minimum-phase equalized channel is obtained, the coefficients of the ECI are copied to an all-pole IIR indirect equalizer that directly inverts the output, of the direct, equalizer. The Routh-Hurwitz criterion [4.6] is introduced for verifying that, the impulse response of the equalized channel is minimum phase.

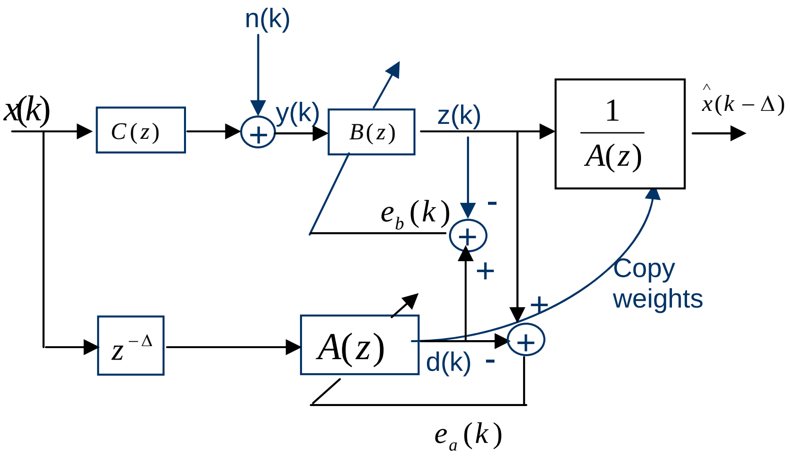


Fig.4.4 Proposed adaptive equalization system

The block diagram of the overall system is shown in Fig. 4.4 is the proposed adaptive IIR equalization system. The data sequence *x (k)* consists of complex, zero-mean, and Gaussian random variable. The channel is characterized by a complex channel response *c(k)* with z-transform *C(z)* and corrupted by independent additive complex white Gaussian noise *n(k)* with variance σ_0^2 .*B(z)* is the system function of an M-tap FIR direct equalizer (DE) and *A(z)* is that of an N-tap FIR equalized channel identifier (ECI). Assuming that the channel impulse response exists only over the time interval [0, *v*T]**,** where T is the sampling period, then the input-output relation for the discrete-time equivalent channel has the form:

$$y(k) = \sum_{m=0}^{v} c(m)x(k-m) + n(k) \tag{4.3}$$

Let's assume an all-zero FIR channel model such that

$$C(z) = \prod_{i=1}^{K}(1-\alpha_i z^{-1})\prod_{j=1}^{L}(1-\beta_j z^{-1}) \tag{4.4}$$

with $|\alpha_i| > 1$ and $|\beta_i| < 1$. Taking partial fraction expansions of l/C(z) and then making it stable impulse response, we obtain

$$\frac{1}{C(z)} = -\sum_{p=0}^{\infty}\tilde{h}_p z^{p+1} + \sum_{q=0}^{\infty}\tilde{h}_q z^{-q} \tag{4.5}$$

where

$$\left\{ \tilde{h}_p = \sum_{k=1}^{K}\left[\prod_{i\neq k}^{K}\frac{1}{(1-\alpha_i\alpha_k^{-1})}\prod_{j=1}^{L}\frac{1}{(1-\beta_j\alpha_k^{-1})}\right]\frac{1}{\alpha_k^{p+1}} \right\}$$

$$\left\{ h_q = \sum_{l=1}^{L} \left[\prod_{i=1}^{K} \frac{1}{(1-\alpha_i \beta_l^{-1})} \prod_{j\neq 1}^{L} \frac{1}{(1-\beta_j \beta_j^{-1})} \right] \beta_l^q \right\} \tag{4.6}$$

From **(**4.6**)** we can conclude that the causal part is caused by ,the minimum-phase components and the anticausal part by the maximum-phase components. Ignoring tail parts of both sides and using a direct equalizer with *M* taps to cancel out the channel effect, the truncated version of (4.5) becomes t he solution of t he direct equalizer, and the causal and anticausal part are centered by the delay of the reference signal.

Configuration of the proposed IIR equalizer

We shall first consider a conventional FIR DE. The case that the ECI is fixed **as** *A(z)* = 1 on adapting the DE corresponds to the conventional DE configuration. In this scheme, an error for updating the DE is defined as

$$e_b(k) = x(k-\Delta) - b^*(k)y(k) \tag{4.7}$$

where

$$\begin{aligned} b^*(k) &= [b_0^*(k), b_1^*(k), \ldots\ldots, b_{M-1}^*], \\ y(k) &= [y(k), y(k-1), \ldots\ldots, y(k-M+1)]^T \end{aligned} \tag{4.8}$$

and $\Delta (0 \le \Delta \le M)$ corresponds to the delay of the DE and superscript "*" denotes the complex-conjugate transpose of a matrix or a vector and the complex conjugate of a scalar.

Direct equalization with a delayed training sequence shows different performance depending on the delay of the filter [4.4], Δ. Letting Δ be half the length of the DE *(M / 2)* would not be optimal but would be quite satisfactory [4.4]. This is due to allocating same number of taps to the anticausal and causal part. The inverse of an unknown channel is divided into anticausal part and causal part with the center of Δ = *M/2*. So, implementing the direct equalization with the reference delay Δ = *M / 2* can achieve the stable performance provided by the sufficient number of taps. However, such direct equalization method is not practical since an actual channel, especially long-delay channel, requires a DE of an infinite number of taps to precisely approximate the inverse of the channel.

To overcome the limit of a DE, the optimal configuration in which an indirect equalizer is combined with the DE is proposed. Obtaining a minimum-phase impulse response of the equalized channel is performed before IIR equalization, thus ensuring stable operation of the IIR equalizer [4.7].

Since all the maximum-phase components of a channel appears at the anticausal part of the inverse of the channel, the anticausal part of the, equalized channel must be removed. For this, we employ a monic causal ECI as a reference generation filter for adaptation of the

DE. Reference sequence provided by the causal ECI causes the DE to be adapted toward suppressing the anticausal part of the equalized channel. Two errors are defined for adaptation of the DE and the ECI, respectively as follows:

$$e_b(k) = a^*(k)x(k-\Delta) - b^*(k)y(k) \tag{4.9}$$

$$e_a(k) = b^*(k)y(k) - a^*(k)x(k-\Delta) \tag{4.10}$$

where

$x(k) = [x(k), x(k-1), \ldots\ldots\ldots, x(k-N+1)]^T$,

$a^*(k) = [a_0^*(k), a_1^*(k), \ldots\ldots\ldots, a_{N-1}^*(k)]$,

The scheme is to minimize $E[|e_b(k)|^2]$ or $E[|e_a(k)|^2]$ with a monic constraints, $a_0(k) = 1$. This constraint prevents the solutions of *a* and *b* from being trivial solutions *a* = *b* = *0.* The optimum solutions are determined at the minimum value of the mean-square error (MSE), which varies, with the value of Δ. The delay Δ determines number of taps allocated to the anticausal and causal parts in the DE. In order for the equalized channel to be minimum phase, the DE approximates at least the inverse of all the maximum phase components of the channel. Since the increase of Δ means the increase of the number of taps of the DE allocated to the anticausal part., Δ is to be its possible maximum value. However, in case of pure minimum-phase channels, the DE requires one tap for causal part. Consequently, considering such many conditions, we select the moderate value of Δ as *M-1***.**

The adaptive algorithm

Global convergence is guaranteed because there is only a single minimum in the quadratic MSE surface. We set the initial coefficients of the DE **as** *b*(0)* = [0,0…0] and those of the ECI as *a*(O)* = [l,0,0,…..,0], which means that the A-delayed training signal becomes the desired signal of the direct equalizer in adaptation of the initial tap coefficients.

The LMS algorithms for adapting the DE and the ECI are given by

$$b(k+1) = b(k) + \mu_b e_b(k) y(k) \tag{4.11}$$

$$a(k+1) = a(k) + \mu_a e_a x(k-\Delta) \tag{4.12}$$

where μ_a and μ_b are the step-sizes which control the convergence speed and the residual error after convergence. In (9), the first element, of *a(k)*, $a_0(k)$ is fixed to unity to satisfy the monic constraint.

The adapted coefficients of the ECI are copied to the fixed indirect IIR equalizer, which will be used for IIR equalization by inverting the equalized channel response. Zero energy anticausal part of the equalized channel certifies to its minimum-phase characteristic i.e. since the inverse of a minimum-phase system is always stable. IIR equalization can be successfully implemented with guaranteed stability.

4.4 Results and Discussion

A random input sequence is passed through the system or channel which is then added with -30dB white Gaussian noise with zero mean and unit variance is used as the training signal, x(k).

Example 1. Channel model 1:

In Fig.4.5, the system transfer function C(z) was taken to be

$$C(z) = (1 - 0.7z^{-13})(1 + 1.3z^{-15}) \tag{4.13}$$

which is a nonminimum phase channel. In addition, 30 dB AWGN exists. M = 113 and N = 15 are used for the DE and the ECI, respectively. For comparison purpose, an FIR equalizer with M = 128 which has the same number of total taps as the proposed IIR equalizer is also implemented with the reference delay of M/2**.** All the adaptive algorithms use the same step-sizes of 0.00005. The square output error, $|x(k) - \hat{x}(k)|^2$ is averaged over 1000 samples and is shown in Fig. 4.5. From the plot we observe that the IIR and FIR equalizers have almost same convergence rate but the IIR equalizer has better performance than the FIR equalizer in terms of residual MSE after convergence.

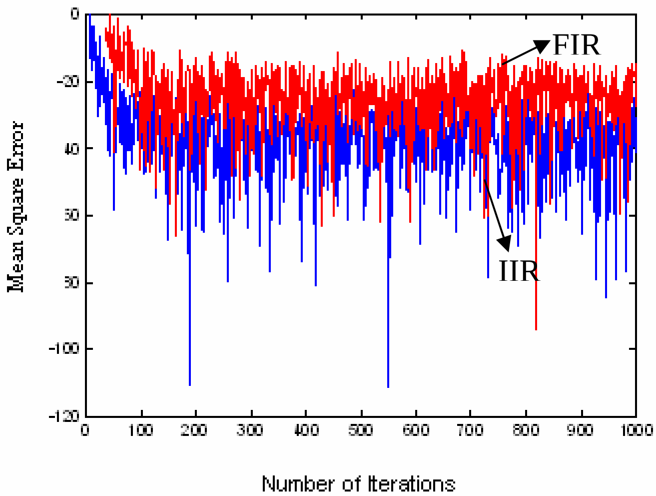


Fig.4.5 Comparison of convergence performance between LMS based adaptive IIR and FIR equalizer

Example 2. Channel model 2:

The transfer function used is given by:

$$C(z) = (1 - 0.81z^{-1})(1 + 0.9z^{-1})(1 + 1.05z^{-1})(1 + 1.15z^{-1}) \tag{4.14}$$

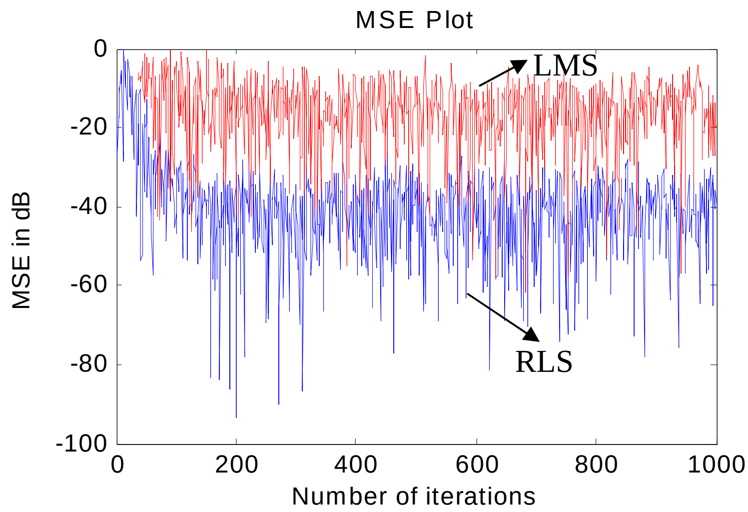


Fig.4.6 Comparison of convergence performance between LMS based and RLS based IIR equalizer

which is also nonminimum phase channel and 30 dB AWGN exists. Since the maximum delay is much smaller than that of the channel 1**,** M = 30 and N = 5 are used for the DE and the ECI, respectively. A step size of 0.001 is used for LMS based adaptive IIR equalizer. From Fig 4.6, it is observed that the convergence performance of the adaptive IIR equalizer using LMS and RLS are different. RLS converges rapidly approximately by 200 iterations and LMS requires approximately 700 iterations to converge.

4.5 Summary

An adaptive IIR equalizer to deal with nonminimum- phase channels with the guaranteed stability. By introducing this above equalization scheme in which the DE and the ECI are combined, we obtained minimum-phase equalized channels. Since an inverse system of any minimum-phase system is always stable, it is possible to implement the IIR equalization with the guaranteed stability. The proposed equalization method provides a near-optimum IIR equalizer that has much better performance than conventional FIR equalizers in terms of convergence speed and residual MSE. It was also given that the performance comparison between the LMS and RLS based adaptive IIR channel equalization. It observed that the RLS algorithm gives superior performance than the LMS convergence.

*****

Chapter 5

SUBBAND ADAPTIVE FILTERING

Multirate signal processing

Analysis of SAF for 2-band case

Results and discussion

CHAPTER 5

5. SUBBAND ADAPTIVE FILTERING

5.1 Introduction

Time domain and transform domain adaptive filtering may be regarded as special cases of subband adaptive filtering. As a new concept, sub-band adaptive filtering was first introduced in the second half of the 1980s [5.13, 5.14]. . Since the introduction of the concept of sub-band adaptive filtering using general multirate structures and filter banks, several new structures are proposed. A new subband adaptive filter using a weighted mean square criterion was proposed in [5.12]. A similar criterion was introduced in [5.15] where a full band adaptive filter is adapted in the subband domain using polyphase decomposition. A new subband adaptive filtering structure with critical sampling was introduced in [5.16]. Subband adaptive filtering is successfully applied in a number of applications, for example adaptive noise cancellation [5.17], acoustic echo cancellation [5.20], and channel equalization [5.18, 5.19]. The main idea in subband adaptive filtering is to split a high order adaptive filtering problem into a number of low order adaptive filters. The general concept of subband adaptive filtering may involve a number of properties, such as improved convergence speed and low computational costs. Normally a combination of both is preferred. Another significant property is low delay, which for example is achieved at the expense of computational complexity.

Different configurations of subband adaptive filters exist. In some configurations, the desired signal is decomposed into subband signals. In [5.21], this is referred to as the open loop configuration, with individual adaptive filters operating in the sub-bands. In the open loop configuration, the adaptive filters are controlled using individual (local) subband error signals, see Fig. 5.1. The open loop approach is also used in [5.10], where a generalized adaptive filtering approach is presented with adaptive cross-filters operating between the subbands. When the open loop approach is used in a system identification scenario, it is shown that cross-filters are necessary to identify the unknown system correctly. The approach with individual filters in the subband, as depicted in Fig. 5.1, has limited performance in terms of full band Mean Square Error. This is essentially the same in a frequency domain adaptive filter, with an individual frequency domain tap update, where circular correlation limits the performance when the gradient constraint is not imposed on the gradient

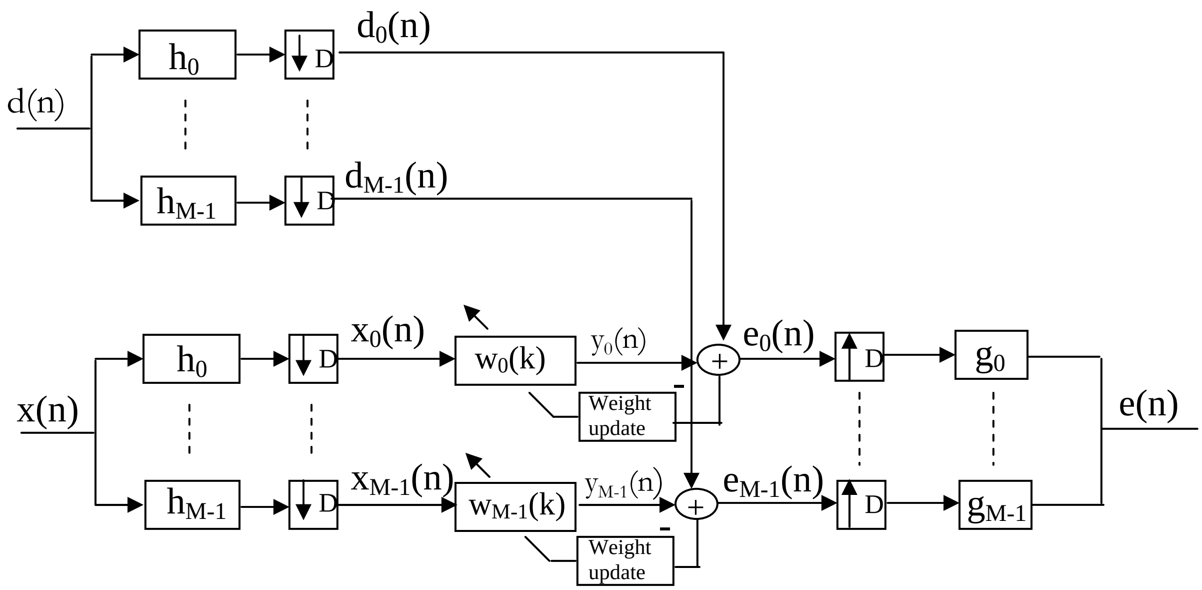


Fig. 5.1 Decomposition of analysis filter output

In another configuration for sub-band adaptive filtering, an error signal is calculated in full band, and decomposed into sub-band error signals, which are used to control the adaptive filters, see Fig. 5.2. In [5.3], this is referred to as the closed loop configuration. In this approach, individual adaptive filters operating in the sub-bands are sufficient for high performance in terms of full band MSE. However, the converge speed may be lower because filter banks impose a signal delay in the control loop.

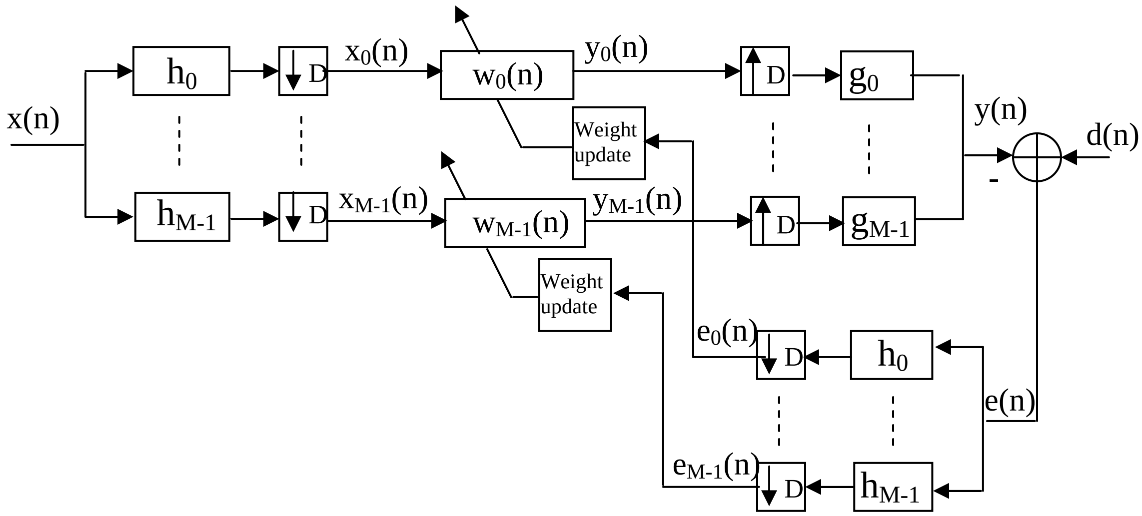


Fig. 5.2 Decomposition of error signal

In a sub-band adaptive filter, the number of sub-bands M , the decimation factor D and the analysis and synthesis filters $H_m(z)$ and $G_m(z)$ are filter bank parameters, which influence the performance of the adaptive filter in terms of full band MSE and convergence speed. An important issue in sub-band adaptive filtering is that multirate filter banks introduce aliasing and imaging distortion, caused by multirate building blocks

5.2 Multirate Signal Processing

In a multirate system [5.4], the signal samples are processed and manipulated at different clock rates at various points in the configuration. Typically the band limited analog signals is sampled at Nyquist rate to generate what we call the fullband signal {x(n)}, with a spectral content from zero to half of sampling frequency . These signal samples can be manipulated by either at higher or lower clock rates by a process called interpolation or decimation. The signals must be proper conditioned by filters prior to or after sampling rate alteration. This operation provides the framework for the sub-band signal decomposition

5.2.1 Decimation and Interpolation

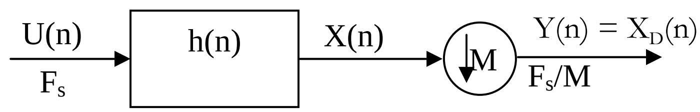


Fig. 5.3 Decimation operation

Decimation is the operation of reducing the sampling rate of a signal by a factor *M*. The process is achieved by passing the full band signal *{U(n)}* through a low pass antialising filter *h(n)* , and then subsampling the filtered signal as shown in Fig. 5.3 . The subsampler is represented by the downsampling arrow and *M* enclosed in the circle .The subsampling process consists of retaining every M^{th} samples of *X(n)* .

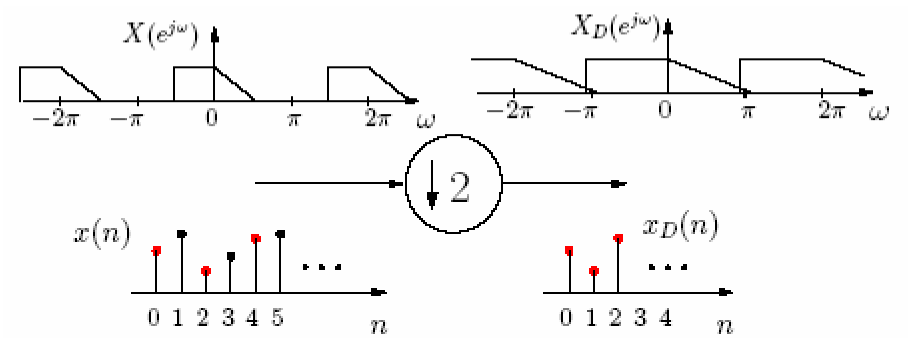


Fig. 5.4 Decimation for M = 2

From Fig. 5.4 we can represent *y(n)* in terms of *x(n)* , as follows

$$y(n) = \begin{cases} x(n) & n = 0, \pm M, \pm 2M \ldots\ldots\ldots\ldots \\ 0, & otherwise \end{cases} \qquad (5.1)$$

In Eq. (5.1) the time scale is compressed by a factor *M*.

It easily follows that

$$Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} X\left(z^{\frac{1}{M}} W^{k}\right) \qquad (5.2)$$

where W = $e^{-\frac{j2\pi}{M}}$

or

$$Y(e^{jw}) = \frac{1}{M}\sum_{k=0}^{M-1} X\left(e^{j\left(\frac{w-2\pi k}{M}\right)}\right) \tag{5.3}$$

Thus the time compression implicit in Eq. (5.1) is accompanied by stretching in the frequency domain so that the interval from 0 to π/M now covers the band from 0 to π .It should be evident that the process of discarding samples can lead to aliasing. To avoid aliasing the bandwidth of full band signal should be reduced to $\pm\ \pi/M$ prior to downsampling by a factor M. This is the function of antialising filter $h(n)$.

Interpolation is the process of increasing the sampling rate of a signal by the integer factor M. As shown in Fig. 5.5 this process is achieved by an upsampler and a low pass filter $g(n)$.

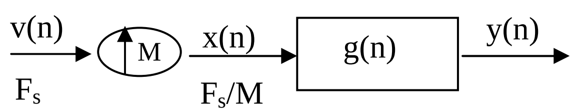


Fig. 5.5 Interpolation operation

The upsampler is shown symbolically in Fig. 5.5 by an upward arrow followed by a filter $g(n)$. This can be mathematically represented as

$$x(n) = \begin{cases} u(n/M) & , n = 0, \pm M, \pm 2M, \pm 3M \ldots\ldots\ldots\ldots\ldots \\ 0 & , Otherwise \end{cases} \tag{5.4}$$

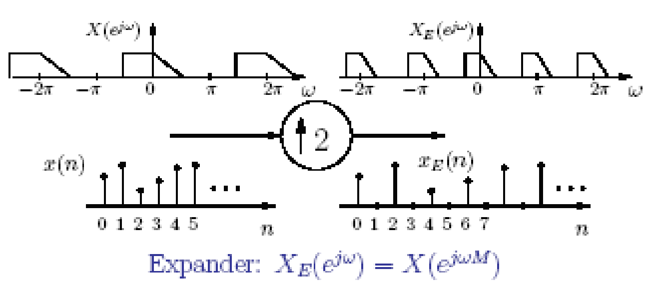


Fig. 5.6 Interpolation for $M = 2$

In Eq. (5.4) the time scale is compressed by a factor *M*.

It easily follows that

$$Y(z) \quad = \quad X(z^{M}) \tag{5.5}$$

Or

$$Y(e^{jw}) \quad = \quad X(e^{jwM}) \tag{5.6}$$

Upsampling has two effects. First stretching the time axis includes compression in frequency; second, forcing the interpolated signal to pass through zero between samples of *x (n)* generates high frequency signals or images.

5.2.2 Polyphase Decomposition

To prevent or reduce the aliasing inherent in the subsampling operator, an antialising filter typically low pass, is usually placed in the front of down sampler as in Fig. 5.7 (a). This combination can be represented by the polyphase decomposition in Fig. 5.7 (b).

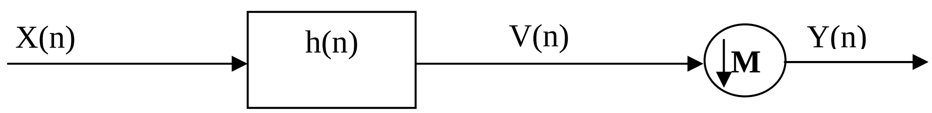


Fig. 5.7 (a) Filter followed by down sampler

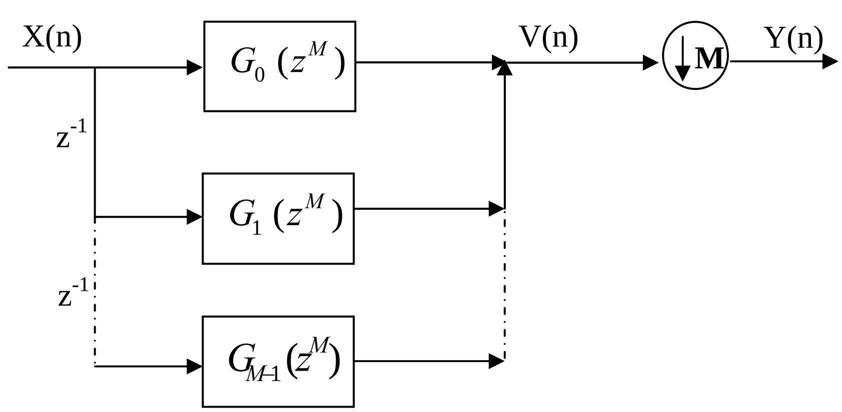


Fig. 5.7(b) Alternative polyphase network presentations

Similarly the polyphase decomposition of the upsampler and filter combination can be demonstrated as follows

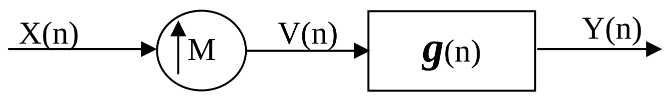


Fig. 5.8 (a) Upsampler followed by filter

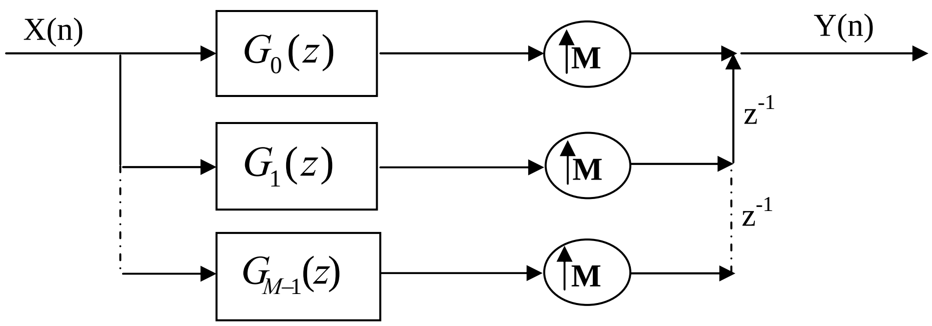


Fig. 5.8 (b) Polyphase decomposition

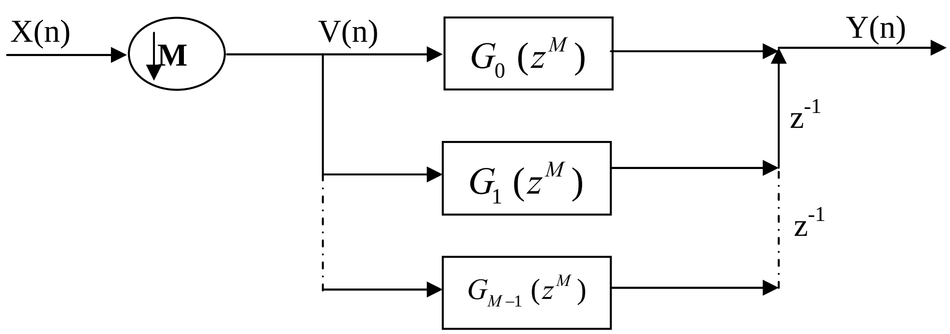


Fig. 5.8 (c) Alternative polyphase network presentations

5.2.3 Two – Channel Filter Bank

The two-channel filter bank provides the beginning of subband coding systems. A encoding bits to each sub-band based on the energy in that band. In this section the requirements and properties of a perfect reconstruction of two-channel subband systems are derived. Analysis filter and synthesis filter with upsampler and downsampler constitutes filterbank. This is represented in Fig. 5.9.

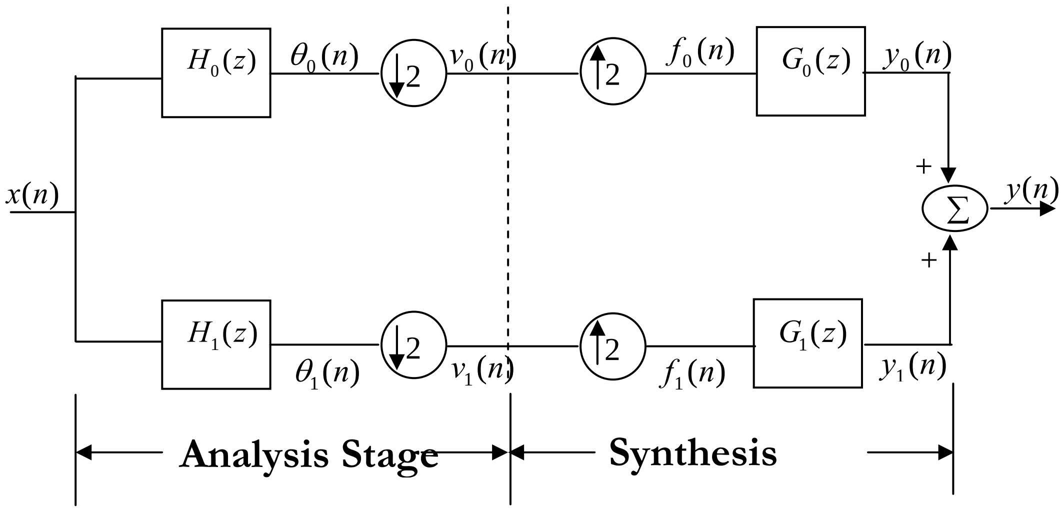


Fig. 5.9 Two- Channel subband filter bank

In Fig. 5.9 the analysis filters are normally low pass and high pass. This structure was introduced in the 1980's. $H_0(z)$ is low pass and $H_1(z)$ is high pass filters . The downsampled signal might be coded for storage or transmission.The effect of using low pass & high pass in analysis part may bring aliasing phenomena, amplitude distortion & phase distortion .The synthesis filters $G_0(z)$ and $G_1(z)$ must be specially adopted to the analysis filters $H_0(z)$ and $H_1(z)$, in order to cancel the errors in this analysis bank. The goal of this section is to discover the conditions for perfect reconstruction as in [5.2, 5.8]. Perfect reconstruction is a crucial property. If the sampling operators and were not present a reconstruction without delay means

$$H_0G_0 + H_1G_1 = I \tag{5.10}$$

A perfect reconstruction with l – step delay without up-sampler and down sampler

$$H_0G_0 + H_1G_1 = z^{-l} \tag{5.11}$$

Tracing the signals through the top branch in Fig. 5.9

$$\begin{aligned} \theta_0(z) &= H_0(z)X(z) \\ Y_0(z) &= G_0(z)F_0(z) \end{aligned} \tag{5.12}$$

As the outputs of the decimation and interpolation filters, while the down-sampler and the upsampler impose, respectively

$$\begin{aligned} V_0(z) &= \frac{1}{2}\left[\theta_0(z^{1/2}) + \theta_0(-z^{1/2})\right] \\ F_0(z) &= V_0(z^2) \end{aligned} \tag{5.13}$$

Combining Eq. (5.12) and (5.13)

$$Y_0(z) = \frac{1}{2}G_0(z)\left[H_0(z)X(z) + H_0(-z)X(-z)\right] \tag{5.14}$$

Similarly,

$$Y_0(z) = \frac{1}{2}G_0(z)\left[H_0(z)X(z) + H_0(-z)X(-z)\right] \tag{5.15}$$

The Z-Transform of reconstructed signal, is then

$$\begin{aligned} X(z) &= \frac{1}{2}\left[H_0(z)G_0(z) + H_1(z)G_1(z)\right]X(z) \\ &\quad + \frac{1}{2}\left[H_0(-z)G_0(z) + H_1(-z)G_1(z)\right]X(-z) \\ &= T(z)X(z) + S(z)X(-z) \end{aligned} \tag{5.16}$$

To eliminate aliasing, we require

$$S(z) = 0$$

This can be achieved by selecting filter co-efficients such that

$$\begin{aligned} G_0(z) &= -H_1(-z) \\ G_1(z) &= H_0(-z) \end{aligned} \tag{5.17}$$

Leaving us with

$$T(z) = \frac{1}{2}\left[H_0(-z)H_1(z) - H_0(z)H_1(-z)\right] \tag{5.18}$$

If overall delay is l for this filter bank, then perfect reconstruction is achievable if

$$G_0(z)H_0(z) + G_1(z)H_1(z) = 2z^{-l} \tag{5.19}$$

And

$$G_0(z)H_0(-z) + G_1(z)H_1(-z) = 0 \tag{5.20}$$

In vector-matrix form these two conditions involve the modulation matrix $H_m(z)$:

$$\begin{bmatrix} G_0(z) & G_1(z) \end{bmatrix} \begin{bmatrix} H_0(z) & H_0(-z) \\ H_1(z) & H_1(-z) \end{bmatrix} = \begin{bmatrix} 2z^{-l} & 0 \end{bmatrix} \tag{5.21}$$

For $l = 0$, i.e. zero delay this matrix can be approximated into

$$\begin{bmatrix} G_0(z) & G_1(z) \end{bmatrix} \begin{bmatrix} H_0(z) & H_0(-z) \\ H_1(z) & H_1(-z) \end{bmatrix} = \begin{bmatrix} 2 & 0 \end{bmatrix} \tag{5.22}$$

The matrix $H_m(z)$ will play a very important role. We can extend this matrix for an M-band filter bank with size $\mathrm{M} \times \mathrm{M}$.

5.3 Analysis of SAF for 2-Band Case

Subband adaptive filtering has been proposed as an alternative for conventional time domain adaptive filtering, [5.5]. The main reason is the reduction in computational complexity and the increase in convergence speed for the adaptive algorithm, which is achieved by dividing the algorithm into subbands. The computational savings comes from the fact that time domain convolution becomes decoupled in the subbands, at a lower sample rate. In these approaches, the underlying signals are decomposed into slightly overlapping frequency bands by passing through a filter bank and the output signals are decimated to give sub-band signals. Now, the adaptations are carried out in each sub-band, but the problem with this approach is the aliasing of the input signals, which arises because of the decimation. Several solutions to this problem, such as over sampling [5.9] of the analysis bank outputs, incorporating adaptive cross filters [5.10] between the adjacent sub-bands, and putting spectral gaps between the bands [5.11], have been recently proposed. In [5.10], it is pointed out that in the M-band adaptive filters with critical sampling, the cross filters can be avoided if the analysis filters are either ideal filters, or the path impulse response is nonzero for the coefficient indices that are multiples of M and zero otherwise. Thus, the cross filters are unavoidable in practical applications. It has been found [5.10] that the convergence performance with the cross filters is not better than that of full band adaptive filter. However, this approach yields a slight gain in computation.

In this Section a new structure for the subband adaptive filter [5.12] with critical sampling and a new criterion for the adaptation algorithm that results in significant improvement in the convergence rate when the LMS algorithm [5.1] is used for adaptation. This structure exploits the polyphase decomposition [5.4] of the adaptive filter. To prevent any distortion that may be introduced in splitting and recombining the signals, we use perfect reconstruction filter banks. All the filters used here are real.

5.3.1 Structures of SAF

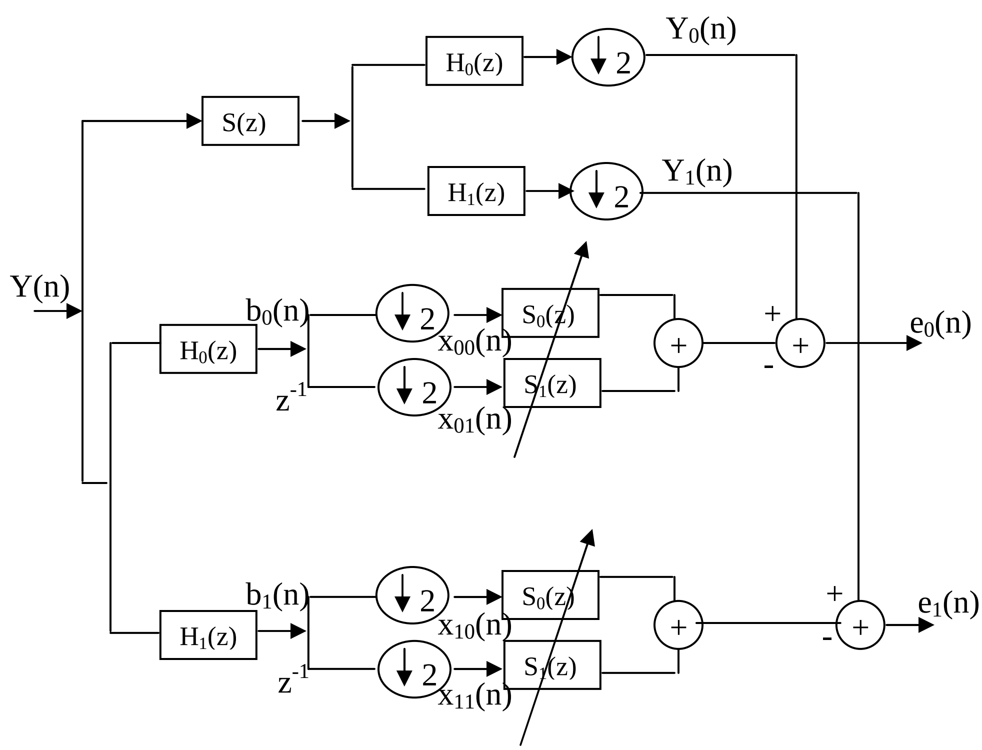


Fig. 5.10 SAF for 2-band case

A structure of the system identification model using SAF [5.12] is given in Fig. 5.10. Here, the output signals from the filters $S(z)$ and $\hat{S}(z)$ are divided into subbands, decimated, subtracted, and combined through an appropriate filter bank to form the error signal *e(n)*. The noise signal is not shown here; we do not consider it in the analysis of the SAF, but its effect on the performance of the SAF will be studied through simulations in later part of this chapter. $H_0(z)$ and $H_1(z)$ are the analysis filters, and $F_0(z)$ and $F_1(z)$ are the synthesis filters. These filters form a perfect reconstruction pair. We used cosine modulated filter-bank [5.9] in our simulation works. The impulse response of the analysis filter $h_k(n)$ are cosine-modulated versions of a prototype filter $h(n)$ of length N_h , and the synthesis filters are obtained via cosine modulation of a length N_f prototype filter $f(n)$. The overall delay D of the filterbank can be fixed arbitrarily in the range $D \in \left[M-1,\ N_f + N_h - M + 1\right]$

For a given delay $D = 2M + d$ (where $0 \le d \le 2M$), the analysis and synthesis filters are given by

$$h_k(n) = 2h(n)\cos\left[\frac{\pi}{M}\left(k+\frac{1}{2}\right)\left(n-\frac{D}{2}\right)+\theta_k\right] \tag{5.23}$$

$$f_k(n) = 2f(n)\cos\left[\frac{\pi}{M}\left(k+\frac{1}{2}\right)\left(n-\frac{D}{2}\right)-\theta_k\right] \tag{5.24}$$

Where $\theta_k = (-1)^k \frac{\pi}{4}$, n = 0,1. …… (M-1). Note that the delay D does not depend upon the filter length but only on the delay of the system. These analysis and synthesis filter gives perfect reconstruction pairs.

As shown in Fig.5.10, $x_{00}(n)$, $x_{01}(n)$, $x_{10}(n)$ and x_{11}(n) are the sub-band components of the input $y(n)$ and together, they account for all the samples of $b_o(n)$ and $b_1(n)$, which are the outputs of the filters $H_0(z)$ and $H_1(z)$ respectively . $S(z)$ and $\hat{S}(z)$ is each of length $L/2$, where L is the length of $\hat{S}(z)$.

5.3.2 Adaptive Algorithm

The $S(z)$ and $\hat{S}(z)$ are to be adopted using the error signals $e_0(n)$ and $e_1(n)$.We have

$$E_0(z) = Y_0(z) - X_{00}(z)\hat{S}_0(z) - X_{01}(z)\hat{S}_1(z) \tag{5.25}$$

$$E_1(z) = Y_1(z) - X_{10}(z)\hat{S}_0(z) - X_{11}(z)\hat{S}_1(z) \tag{5.26}$$

Assume that the filters $H_0(z)$ and $H_1(z)$ of equal bandwidth. If the spectrum of $y(n)$ is flat, then the power of $b_o(n)$ and $b_1(n)$ will be equal . But, if the input is a colored one, then these powers will be different. The cost function is defined as

$$J(n) = E\left[\alpha_0 e_0^2(n) + \alpha_1 e_1^2(n)\right] \quad (5.27)$$

where α_0 and α_1 are inverse of powers of $b_o(n)$ and $b_1(n)$, respectively .This cost function brings down the eigen value spread of the weighted sum of the correlation matrix of the input signals to the adaptive filter , thereby resulting improved rate of convergence . The gradient-based algorithm for adaptation is given by

$$\hat{s}_{0k}(n+1) = \hat{s}_{0k}(n) - \mu \frac{\partial J}{\partial \hat{s}_{0k}} \quad (5.28)$$

$$\hat{s}_{1k}(n+1) = \hat{s}_{n+1}(n) - \mu \frac{\partial J}{\partial \hat{s}_{1k}} \quad (5.29)$$

$$k = 0, 1, 2 \ldots (L/2-1)$$

Now replacing the true gradient by instantaneous gradient the update equation will be as follows

$$\hat{s}_{0k}(n+1) = \hat{s}_{0k}(n) + 2\mu\left[\alpha_0 e_0(n) x_{00}(n-k) + \alpha_1 e_1(n) x_{10}(n-k)\right] \quad (5.30)$$

$$\hat{s}_{1k}(n+1) = \hat{s}_{1k}(n) + 2\mu\left[\alpha_0 e_0(n) x_{01}(n-k) + \alpha_1 e_1(n) x_{11}(n-k)\right] \quad (5.31)$$

These are the LMS adaptation equations for the co-efficients of $\hat{S}_0(z)$ and $\hat{S}_1(z)$ in the two sub-bands are constrained to be the same.

5.4 Results & Discussion

In this section, we study the convergence performance of the SAF using computer simulations. The input signal is a first-order autoregressive (AR) process with white Gaussian noise as the driving input. That is, *y(n)* is modeled as $y(n) = \rho y(n-1) + u(n)$, where *u(n)* is a white Gaussian noise sequence. In our simulations, we fixed ρ at 0.9. The system noise is a white Gaussian noise sequence that is independent of the *u(n)*.

We normalized the input *y(n)* such that the variance of the resulting sequence at the output of *S(z)* was unity. In this case, i.e., for L=80, we considered two levels of system noise: no noise and 20 dB noises. In the simulations, we discarded the first 2000 samples of so that the actual AR sequence used was nearly stationary. The coefficients $\{\alpha_k\}$ were computed as the inverse of the powers of $\{b_k(n)\}$ estimated from the overall samples used in the adaptation.

In the fullband case, we used a normalized LMS algorithm, whereas in the sub-band case, the algorithm as given by Eq. (5.28) and (5.29) was implemented, initializing the coefficients of zero in each case. The best possible value for (best in the sense that it yields fastest convergence with the converged value as close to the noise level as possible) was found by trial and error. Note that the value of μ so found depends on the value of M .The norm and mean square error (MSE) curves were averaged over 50 independent computer runs. The delays introduced by the analysis bank alone and that introduced by the cascade of analysis and synthesis banks are taken into account while plotting the coefficient error vector norm and MSE curves, respectively. That is, the norm and MSE curves are plotted without the effect of the filter bank delay. We may point out Here that for the case of $L=80$, the lengths of the analysis filters (as well as the synthesis filters) were increased with so that the ratio of the transition band to the passband was maintained nearly the same for all values of M .In particular, we used filters (analysis and synthesis) with lengths 20 for $M = 2$.

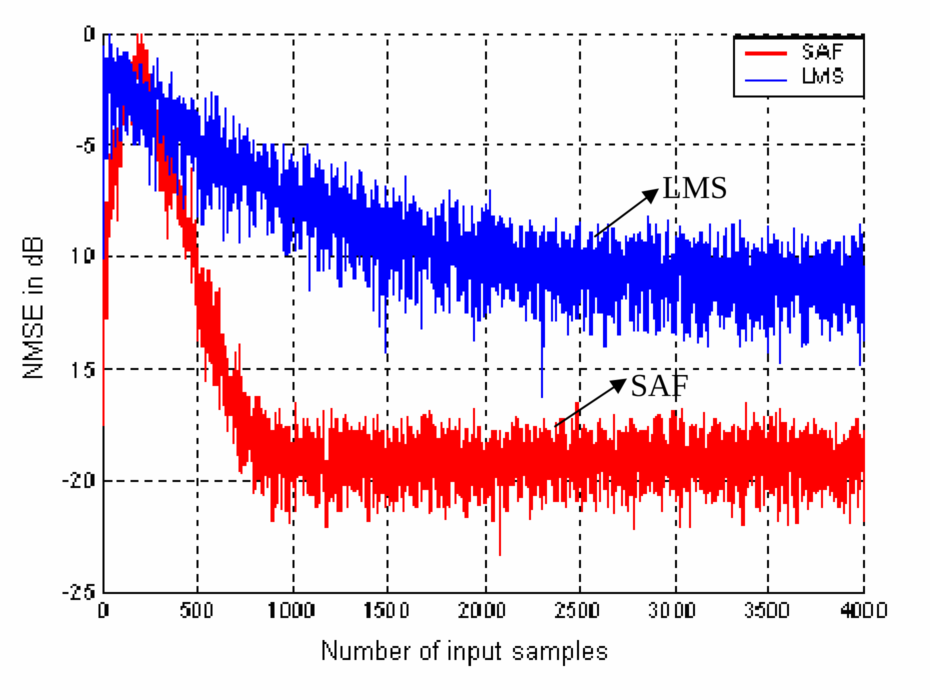


Fig. 5.11 Convergence performance of the SAF for M=2 with system noise absent (filter length L=80), No of training samples = 4,000

. Fig. 5.11 presents the performance of fullband and subband with no noise level. The curve depicts significant increase in convergence rate when the no. of bands was increased. In this, for a given number of training samples the LMS converges to approximately -12dB where as the SAF converges approximately -19dB for less than the given number of samples used for LMS counterpart.

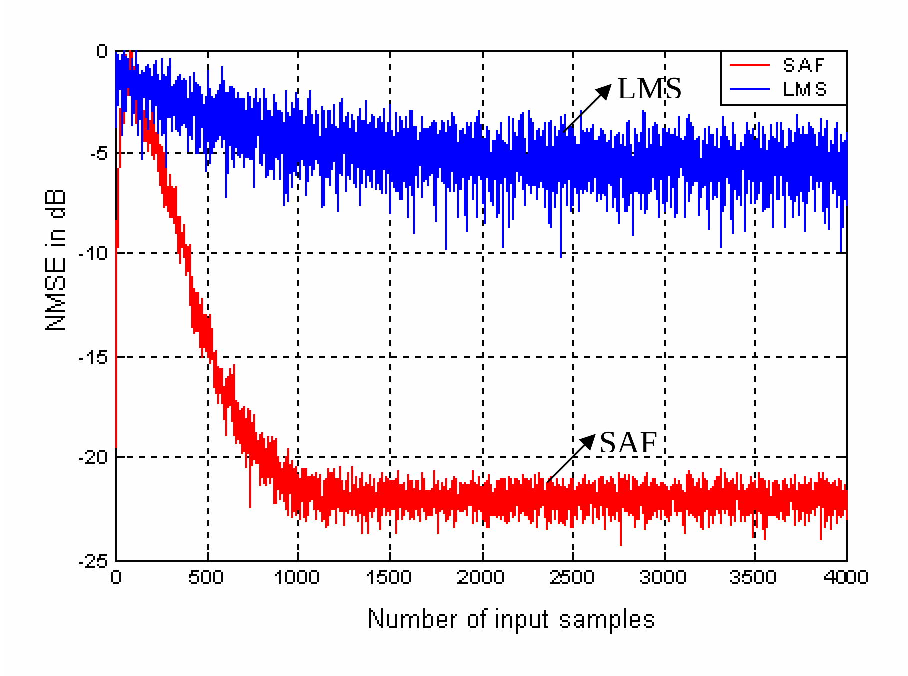


Fig. 5.12 Convergence performance of the SAF for M=2 with system noise level of –20 dB (filter length L=80), No of training samples = 4,000

Similarly in Fig. 5.12, we compared the performance and convergence rate with 20dB noise level. The curves show that the co-efficient error vector norm converges to about 7dB above the system noise level in the subband case, whereas 23dB in case of fullband.

5.5 Summary

A new structure for the subband adaptive filter is presented. The convergence rate of the SAF improves considerably with the increase in no. of bands. The cross filters are totally avoided in the structure and the adaptive filters in the subbands are independent of the analysis and synthesis filters. There is no such significant improvement in the computational complexity as compared to fullband, but a significant improvement comes into picture in terms of learning rate. The agreement of SAF based response is better than that obtained from fullband case. Exhaustive simulation study is further required to demonstrate the superiority of SAF over fullband both for system identification and channel equalization tasks.

*****

Chapter 6

CONCLUSION AND FUTURE SCOPE

CHAPTER 6

6. CONCLUSION AND FUTURE SCOPE

In this thesis, we have made an extensive study of estimation of adaptive IIR filter behavior in signal processing and communication channels. The objective of this research is to evaluate the performance of adaptive IIR system identification and channel equalization using numerous algorithms such as LMS, RLS, GA and SAF. The principle of adaptive IIR filtering process is focused which has numerous applications in various areas.

It was stated that the RLS based adaptive IIR system identification gives superior performance than the LMS counterpart. The proposed equalization method provides a near-optimum IIR equalizer that has much better performance than conventional FIR equalizers in terms of convergence speed and residual MSE. It was also given that the performance comparison between the LMS and RLS based adaptive IIR channel equalization and observed that the RLS based IIR equalizer gives superior performance than the LMS convergence. Due to the multimodal error surface of IIR filtering, special attention has to be given to the learning algorithms in order to avoid the local minimum problem. The gradient-descent algorithms can be affected by a cost surface local minimum. The algorithm is therefore affected by the initialization and can produce inconsistent results. The GA is not a gradient-descent algorithm and is not limited by a local cost surface condition. This training algorithm can therefore produce a more consistent performance result. Thus GA does provide a useful strategy that has quick convergence characteristics. The SAF structure identifies a given system in the subband domains; as a result faster convergence is obtained. The subband filters are updated individually within their own frequency range. Even though the adaptive structure more complex than the fullband case, it is a novel identification method.

It is anticipated that future research will focus on other forms of evolutionary computing techniques like Particle Swamp Optimization, Ant Colony Optimization and Bacteria Foraging are used both for the adaptive IIR system identification and channel equalization applications. The computational overhead with GA is very high. Hence research needs done to reduce the computational complexity of this algorithm.

*****

References

REFERENCES

[1.1] Ng S., Leung S., Chung C, Luk A., Lau W., "The genetic search approach", IEEE signal processing magazine, Volume.13, No.6, (Nov. 1996): pp.38-46.

[1.2] Goldberg D. Genetic algorithms in search optimization. Addison-Wesley, 1989.

[1.3] Itoh Y., "An acoustic echo canceller for teleconference," in Proc. IEEE ICC, (1985): pp. 1498-1502.

[1.4] Pradhan S. S., Reddy V. U., "A New Approach to Subband Adaptive Filtering," IEEE Trans on Signal Processing, Volume 47, (March 1999): pp.655–664.

[1.5] Widrow B., Stearns S. D., **"**Adaptive Signal Processing", Singapore:Second Edition, Pearson,2001

[1.6] Johnson C. R. Jr., "Adaptive IIR filtering: Current results and open issues," IEEE Trans. Inform. Theory, Volume IT-30, No. 2, (Mar. 1984): pp. 237-250.

[1.7] [1.7] White S. A., "An adaptive recursive filter," in Proc. 9th Asilomar Conf. Circuits, Syst., Comput., Pacific Grove, CA, (Nov. 1975): pp. 21-25.

[2.1] Proakis John G., Manolakis Dimitris G., Digital Signal Processing.New Delhi: Prentice-Hall, 2004

[2.2] Ljung L., System Identification. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[2.3] Haykin Simon., Adaptive Filter Theory, ,Pearson Education Publisher, 2003.

[2.4] Fan H., Jenkins W.K., "A New Adaptive IIR Filter," IEEE Trans. Circuits and Systems, Volume. CAS-33, no. 10, (Oct. 1986): pp. 939-947.

[2.5] Ho K.C. and Chan Y.T., "Bias Removal in Equation-Error Adaptive IIR Filters", IEEE Trans. Signal Processing, Volume. 43, no. 1, (Jan. 1995): pp.1054-1068

[2.6] Cousseau J.E., and Diniz P.S.R., "New Adaptive IIR Filtering Algorithms Based on the Steiglitz-McBride Method", IEEE Trans Signal Processing, Volume. 45, no. 5, (May 1997).

[2.7] Haykin Simon, Adaptive Filter Theory, Upper Saddle River, NJ, Third Edition, Prentice-Hall Inc, 1996.

[2.8] Landau I. D., Adaptive Control: The Model Reference Approach. Marcel Dekker, New York, 1979.

[2.9] Lucky R.W., "Soft constraint satisfaction (SCS) blind channel equalization algorithms" Wiley Interscienc , Volume 12, Issue 2,(1998): pp.117 – 134.

[2.10] Qureshi S.U.H., "Adaptive Equalization", Proc. IEEE, Volume.73,no.9, (Sept.1985): pp.1349-1387.

[2.11] Abreu E., Mitra S.K., and Marchesani R., "Non minimum phase channelequalization using non-causal filters", IEEE Trans.Signal Processing, Volume.45, no.30, (1997): pp.1-13

[3.1] Ljung L., System Identification. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[3.2] Mosquera Carlos, Lopez Roberto, and Perez Fernando, "On the Bias of the Modified Output Error Algorithm" IEEE transactions on automatic control, Volume. 43, no. 9, (September 1998).

[3.3] Landau I. D., Adaptive Control: The Model Reference Approach. Marcel Dekker, New York, 1979.

[3.4] Widrow and Stearns S. D., Adaptive Signal Processing. Englewood Cliffs, N.J.: Prentice-Hall, 1985.

[3.5] Widrow B., McCool J.M., Larimore M.C., and Johnson C. R., Jr., "Stationary and nonstationary learning characteristics of the LMS adaptive filter," Proc IEEE, Volume. 64, no. 8, (Aug. 1976): pp. 1151-1162.

[3.6] Mosteller F. and Tukey J. W., Data Analysis and Regression. Addison-Wesley Reading, Mass., 1977.

[3.7] Mendel J. M., Lessons in Digital Estimation Theory. Prentice-Hall, Englewood Cliffs, N.J., 1987.

[3.8] Haykin S. S., Adaptive filter Theory. Prentice-Hall, Englewood Cliffs, N. J.,1986.

[3.9] Ljung L. and Soderstrom T., Theory and Practice of Recursive Identification.Cambridge, MIT Press, 1983.

[3.10] Landau I**.** D., Adaptive Control: The Model Reference Approach. New York, Marcel Dekker, 1979.

[3.11] Stearns S. D., “Error surfaces of recursive adaptive filters,” IEEE Trans.Circuits Systems, Volume. CAS-28, no. 6, (June 1981):pp. 603-606.

[3.12] Levinson Wiener, “A refined Wiener-Levinson method in frequency analysis Source,” SIAM Journal on Mathematical Analysis archive ,Volume 27 , (Sept 1996): Issue 5

[3.13] Shynk J. J., "Adaptive IIR Filtering", IEEE Acoustic Speech and SignalProcessing Magazine, Volume 6,(April 1989): pp. 4-21.

[3.14] Stoica P. and Sderstrom T., "The Steiglitz-McBride identification algorithm revisited-Convergence analysis and accuracy aspects", IEEE Trans. On Automatic Control, Volume 26**,** (June 1981): pp. 712-717.

[3.15] NG. S.C, Leung S.H., Chung C.Y, Luk. A and Lau W.H, “The Genetic Search Approach”, IEEE Signal Processing Magazine,Nov.1996.

[3.16] Larimore M.G.,”A new HARF algorithm with adaptive SPR-Circuits and Systems”, IEEE. Trans., Volume.ASSP-28, (1980): pp.428-440.

[3.17] Fogel D., “What is evolutionary computing”, IEEE spectrum magazine, (Feb. 2000): pp.26-32.

[3.18] Goldberg D., Genetic algorithms in search optimization, Addison-Wesley, 1989

[3.19] Chung T., Leung H., “A genetic algorithm approach in optimal capacitor selection with harmonic distortion considerations”, International Journal of Electrical Power & Energy Systems, Volume.21, no.8, (Nov. 1999) : pp.561-569.

[3.20] Jiabao G., Aral M., “Progressive genetic algorithm for solution of optimization problems with non-linear equality and inequality constraints”, Applied Mathematical Modeling, Volume.23, no.4, (April 1999): pp.329-43.

[3.21] Lei X, Lerch E, Povh D. “Genetic algorithm solution to economic dispatch problems”, European Transactions on Electrical Power, Volume.9, no.6, (Nov.-Dec 1999) : pp.347-353.

[3.22] Peigin S., Mantel B., Periaux J., Timchnenko S., Borodin A., “Application of agenetic algorithm to a heat flux optimization proble”, Surveys on Mathematics for Industry, Volume.9, no.3, (2000) :pp.235-45.

[3.23] Rong Y., “Solving large travelling salesman problems with small populations”, Second International Conference on Genetic Algorithms in Engineering Systems:Innovations and Applications, IEE, (1997):pp.157-62.

[3.24] Sayoud H., Takahashi K., Vaillant B., "Minimum cost topology optimization of ATM networks using genetic algorithms", Electronics Letters, Volume.36, no.2, (Nov .2000): pp.2051-2063.

[3.25] Ng S., Leung S., Chung C, Luk A., Lau W., "The genetic search approach", IEEE signal processing magazine, Volume.13, no.6, (Nov. 1996) : pp.38-46.

[3.26] Ma Q., The application of genetic algorithms to the adaptation of IIR filters, PhD. Thesis, Department of electrical and electronic engineering, Loughborough University of Technology, Nov. 1995.

[4.1] Qureshi S., "Adaptive equalization", Proceedings of the IEEE, Volume.73, no.9, (Sept. 1985): pp.1349-1387.

[4.2] Chen S., Mulgrew B., McLaughlin S., "Adaptive Bayesian Equalizer with Decision Feedback", IEEE Trans Signal Processing, Volume. 41, no.9, (Sept. 1993): pp.2918-2927.

[4.3] Proakis J.G. , Digital Communication, New York: McGraw-Hill. 1983.

[4.4] Kim Hyoung-Nam and Song Woo- Jin,"An adaptive IIR equalizer for nonminimum-phase channels", Proceedings of ICSP, volume.1, (1998): pp:441 – 444.

[4.5] Abreu E., Mitra S.K., and Marchesani R. "Non-minimum phase channel equalization using non-causal filters," IEEE Trans. Signal Processing, Volume. 45, no.30, (1997): pp.1-13.

[4.6] Chen C. T., "Linear System Theory and Design", HRW Series in Electrical and Computer Engineering, pp.412-424.

[4.7] S.C. Ng, C.Y. Chung, S.H. Leung, and Andrew Luk, "An Evolutionary Search Algorithm for Adaptive IIR Equalizer", Proc. ZEEE International Symposium on Circuits and Systems, London, UK, Volume. 2, (May 94): pp. 53-56.

[5.1] Bernard Widrow & Samuel D.Streans. Adaptive Signal Processing, Pearson Education Publisher,2001

[5.2] Gilbert Strang & Traung Nguyen, Wavelets and Filter Banks, Wellesley – Cambridge Press.

[5.3] W. Kellermann, "Analysis and design of multirate systems for cancellation of acoustical echoes," Proc. IEEE ICASSP, (1988): pp. 2570-2573.

[5.4] A.N.Akansu & R.A. Haddad, Multiresolution Signal Decomposition, Academic Press.

[5.5] Proakis & Monalakis, Digital Signal Processing, Prentice Hall Of India, 2004.

[5.6] Gilloire A. and Vetterli M., "Adaptive Filtering in Subbands," IEEE International Conference on Acoustics, Speech and Signal Processing,Volume. 3, (1988):pp. 1572–1575.

[5.7] Nieemisto R. & Tabus I.**,** "Signal Adaptive Sub band Decomposition for Adaptive Noise Cancellation", ECCTD, 2001

[5.8] Vaidyanathan P. P., Multirate Systems and Filter Banks. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[5.9] Heller P.N. and Nguyen T.Q., "A general formulation of modulated filter banks," IEEE Trans on Signal Processing, Volume. 47, no. 4, (April 1999): pp. 986- 1002.

[5.10] Gilloire A. and Vetterli M., "Adaptive Filtering in Subbands with Critical-Adaptive Filtering in Subbands with Critical Sampling: Analysis Experiments, and Application to Acoustic Echo Cancellation," IEEE Trans on Signal Processing, vol. 40, no. 8, (August 1992): pp. 1862–1875.

[5.11] Yasukawa H. and Shimada S., "Acoustic Echo Canceller with High Speed Quality," IEEE International Conference on Acoustics, Speech and Signal Processing, (1987): pp. 2125–2128.

[5.12] Pradhan S. S. and Reddy V. U., "A New Approach to Subband Adaptive Filtering,"IEEE Trans on Signal Processing, Volume. 47, (March 1999): pp. 655

[5.13] Chen J., Bes H., Vandewalle J., and Janssens P., "A New Structure for Sampling: Analysis, Experiments, and Application to Acoustic Echo Cancellation," IEEE Trans on Signal Processing, Volume. 40, no. 8, (August 1992) : pp. 1862–1875.

[5.14] Ono Y. and Kiya H., "Performance Analysis of Subband Adaptive System using an Equivalent Model," IEEE International Conference on Acoustics, Speech and Signal Processing, Volume. 3, (1994) :pp. 53–56.

[5.15] Courville M. de and Duhamel P., "Adaptive Filtering in Subbands Using Weighted Criterion," IEEE Trans. on Signal Processing, Volume. 46, no. 9, (September 1998): pp. 2359–2371.

[5.16] Petraglia M. R., Alves R. G., and Diniz P. S. R., "New Structures for Adaptive Filtering in Subbands with Critical Sampling," IEEE Trans. on

Signal Processing, Volume. 48, no. 12, (December 2000): pp. 3316–3327.

[5.17] H. R. Abutalebi, H. Sheikhzadeh, R. L. Brennan, and G. H. Freeman, "A Hybrid Subband Adaptive System for Speech Enhancement in Diffuse Noise Fields," IEEE Signal Processing Letters, Volume. 11, no. 1, (2004): pp. 44–47.

[5.18] Nordberg J., Blind Subband Adaptive Equalization**,** Blekinge Institute of Technology Research Report 2002/11, ISSN 1103-1571, 2002.

[5.19] Weiss S., Dooley S. R., R. W. Stewart, and Nandi A. K., "Adaptive Equalization in Oversampled Subbands," Conference Record of the Thirty-Second Asilomar Conference on Signals, Systems and Computers, Volume. 1, (November 1998): pp. 389–393.

[5.20] Breining C., Dreiseitel P., Hansler E., Mader A., Nitsch B., Puder H., Schertler T., Schmidt G., and J. Tilp, "Acoustic Echo Control - An Application of Very-High-Order Adaptive Filters," IEEE Signal Processing Magazine, (July 1999): pp. 42–69.

[5.21] Morgan D. R. and Thi J. C., "A Delayless Subband Adaptive Filter Architecture," IEEE Trans. on Signal Processing, (August 1995): pp. 1819–1830

*****